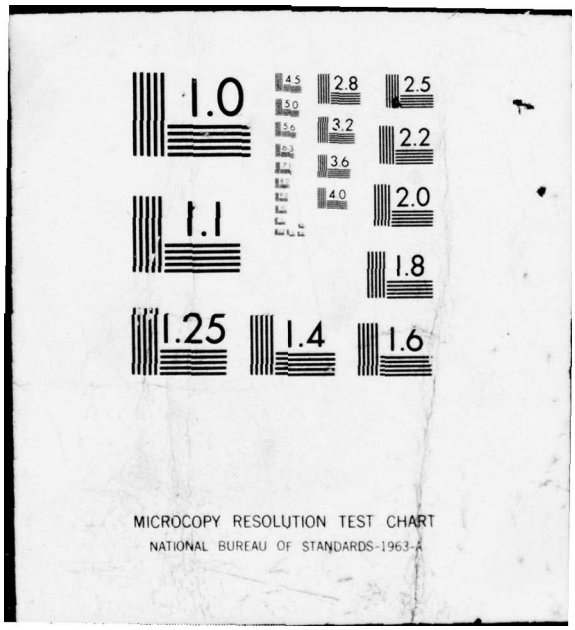
AD-A031 061 ALABAMA UNIV IN HUNTSVILLE DEPT OF ELECTRICAL ENGINEERING F/G 9/2
A MICROPROCESSOR SUBSYSTEM FOR AUTOMATIC TESTING.(U)
AUG 76 D K FRONEK, D G GREEN DAAG29-76-G-0062

UNCLASSIFIED ARO-13594.1-R-EL NL

1 OF 1
AD
A031061

END
DATE FILMED
11-76


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

13

FINAL REPORT

U. S. Army Research Office
Research Triangle Park, North Carolina 27709

Grant Number DAAG29-76-G-0062
Completed on 14 June 1976

A MICROPROCESSOR SUBSYSTEM FOR AUTOMATIC TESTING

Department of Electrical Engineering
The University of Alabama in Huntsville

August 1976

Donald K. Fronek
Assistant Professor

David G. Green
Research Engineer

FINAL REPORT

U. S. Army Research Office
Research Triangle Park, North Carolina 27709

Grant Number DAAG29-76-G-0062

Completed on 14 June 1976

A MICROPROCESSOR SUBSYSTEM FOR AUTOMATIC TESTING

Department of Electrical Engineering
The University of Alabama in Huntsville

August 1976

Donald K. Fronek
Assistant Professor

David G. Green
Research Engineer

"A MICROPROCESSOR SUBSYSTEM FOR AUTOMATIC TESTING"

INTRODUCTION

This research results from the 1975 Summer Laboratory Research Cooperative Program (LRCP) conducted at the U. S. Army Missile Command, Redstone Arsenal, Alabama. (Task Order 75-232) The Task Order was concerned with investigating the feasibility of developing a microprocessor that would perform arithmetic and logic functions not being performed within the Land Combat Support System (LCSS) automated test equipment. This additional capability will enhance many of the measurement administrative functions currently being done by analog techniques. Furthermore, the speed of execution of the punched tape program used with various Units Under Test (UUT) would be enhanced. A microprocessor system would also provide an excellent opportunity for operator designed programs for "debug" and software maintenance. The following block diagram, Figure 1, indicates the current relationship of the microprocessor subsystem relative to the LCSS.

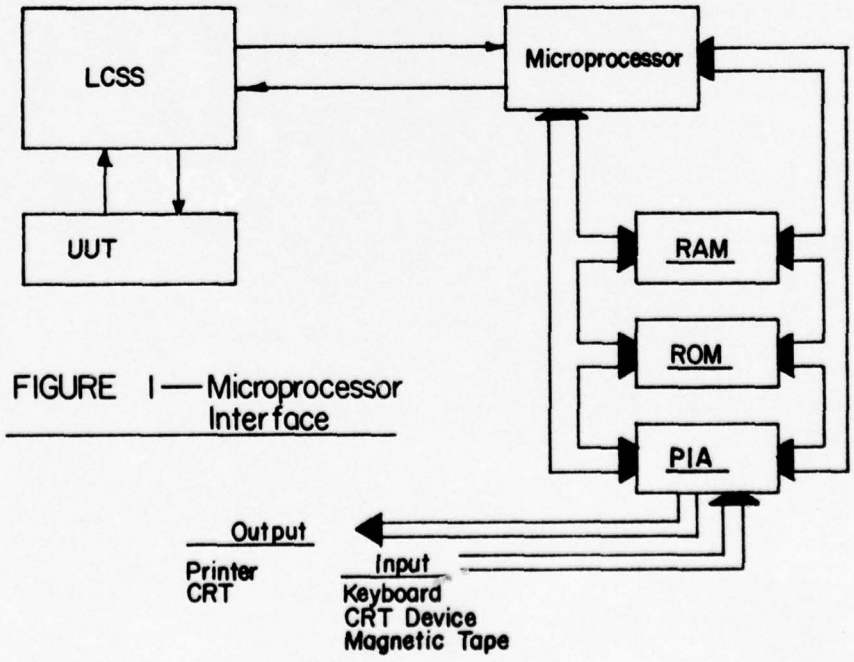


FIGURE 1 — Microprocessor Interface

TECHNICAL OBJECTIVES

The research effort has progressed in four parts:

1. Study existing and projected measurement requirements of the LCSS relative to a resident microprocessor.
2. Design and fabricate a microprocessor system capable of controlling the administrative and logic functions of the LCSS.
3. Develop corresponding software subroutines to support item 2 above.
4. Interface the microprocessor and perform functional testing.

ACCOMPLISHED RESEARCH

This research can be adequately described in two categories - hardware and software. The hardware portion of this Report relates that effort of providing diagrams, descriptions, and prototype models necessary to execute the program as specified by the user. The software that is provided compliments the hardware and provides the execution of logic and arithmetic functions under program control. These large categories will be discussed in detail.

HARDWARE

The microprocessor integrated circuit is a logic device capable of providing addition, subtraction, shifting, transfers, and so on. This powerful device is supported by additional microcomputer elements such as Read Only Memory, ROM, Random Access Memory, RAM, and Peripheral Interface Adapter, PIA, circuit chips. The first problem in this research was to develop a working microprocessor system so that suitable software could be developed. This was no easy task. At least 75% of the total time during the research period was devoted to this effort. This developmental microprocessor is shown in Figure 2. As can be seen, certain control inputs are provided on the front panel -- Reset, Non-maskable Interrupt, (NMI), Halt, and so on. An Address and Data Register

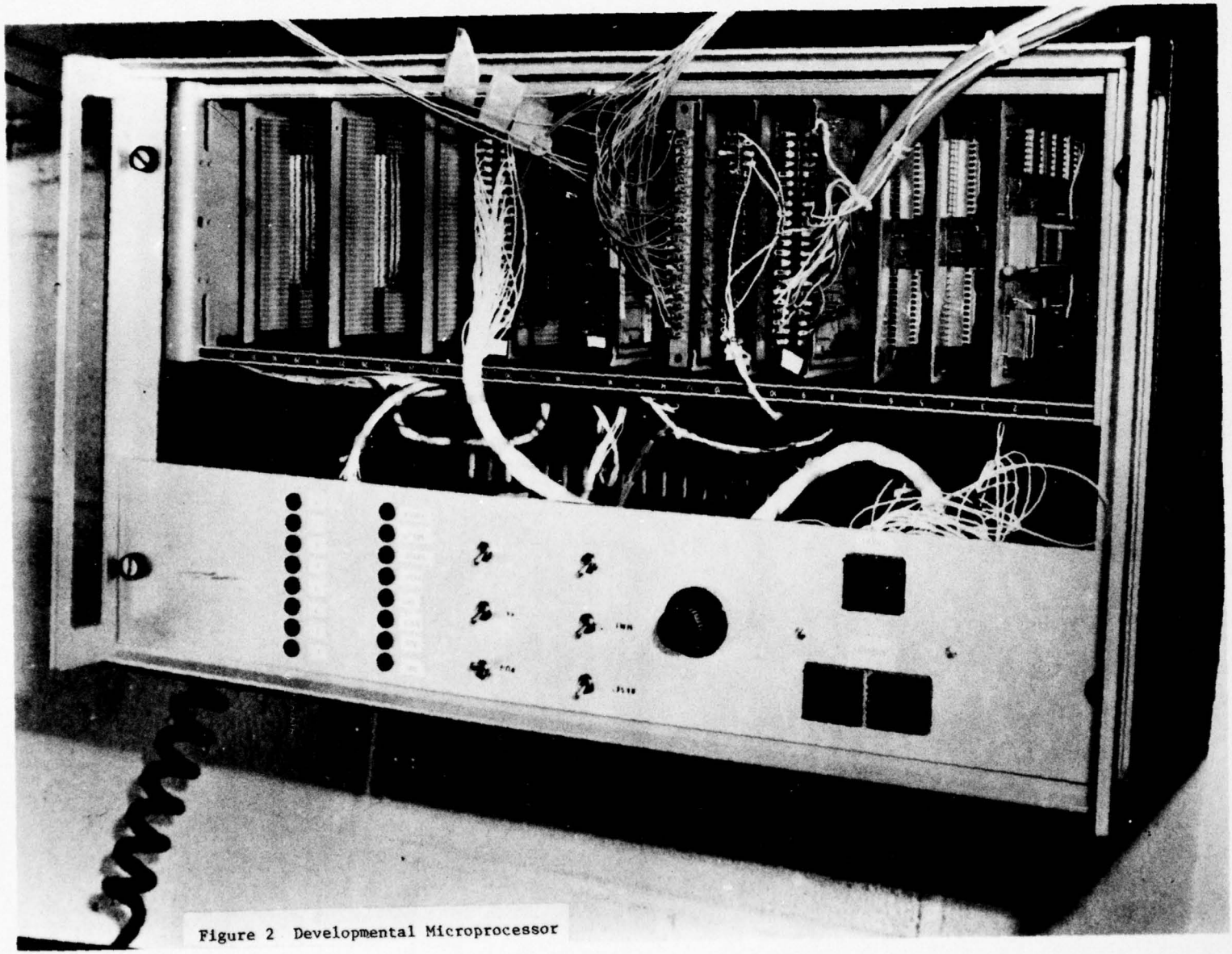


Figure 2 Developmental Microprocessor

display is provided for single step operation. The bulk of the system resides in the printed circuit card frame located in the bottom portion of the cabinet. Each printed circuit board card supports the microprocessor in some fashion -- Interface, Control, Memory, etc.

The input and output devices for this project were limited to

1. Teletype
2. High speed paper tape reader
3. Video displayed administrative text
4. A microprocessor simulator (KIM-1A Microprocessor)

Since the results of this research were to provide information leading to the production of a microcomputer to be used with the LCSS, the computer components would be fabricated using printed circuit boards. The printed circuit boards would provide a modular approach to expansion and addition of peripheral interface circuits. All circuit diagrams in this Report have been placed in the U. S. Army Missile Command System of Drawing, Redstone Arsenal, Alabama. Figure 3 (PC Number 200035-4A) illustrates the completed design of the microprocessor, the microprocessor RAM stack, the MPU Clock, the MIKBUG operating ROM chip and the necessary input control circuitry (drawing number 18876-200035). The 44-pin connector provides transfer of signals to the bus structure located on the card rack back plane. The address of the MIKBUG operating ROM (MCM6830L7) is located at E000 (a hexadecimal notation). The RAM stack (MCM6810L) is located at address A000 (Hex). Both of these address locations are specified by the MIKBUG operating system. The opearting system provides the necessary software subroutines for loading the microprocessor, interagating and changing memory, printing the contents of the MPU, and executing the users program. These operating subroutines are discussed adequately in Motorola Application note 100 [1].

The basic interface circuit is the teletype peripheral adapter card shown in Figure 4. This circuit board was developed to interface a standard TTY current loop or a RC-232 standard bus connection. This card contains optical isolators and the rate generator necessary for

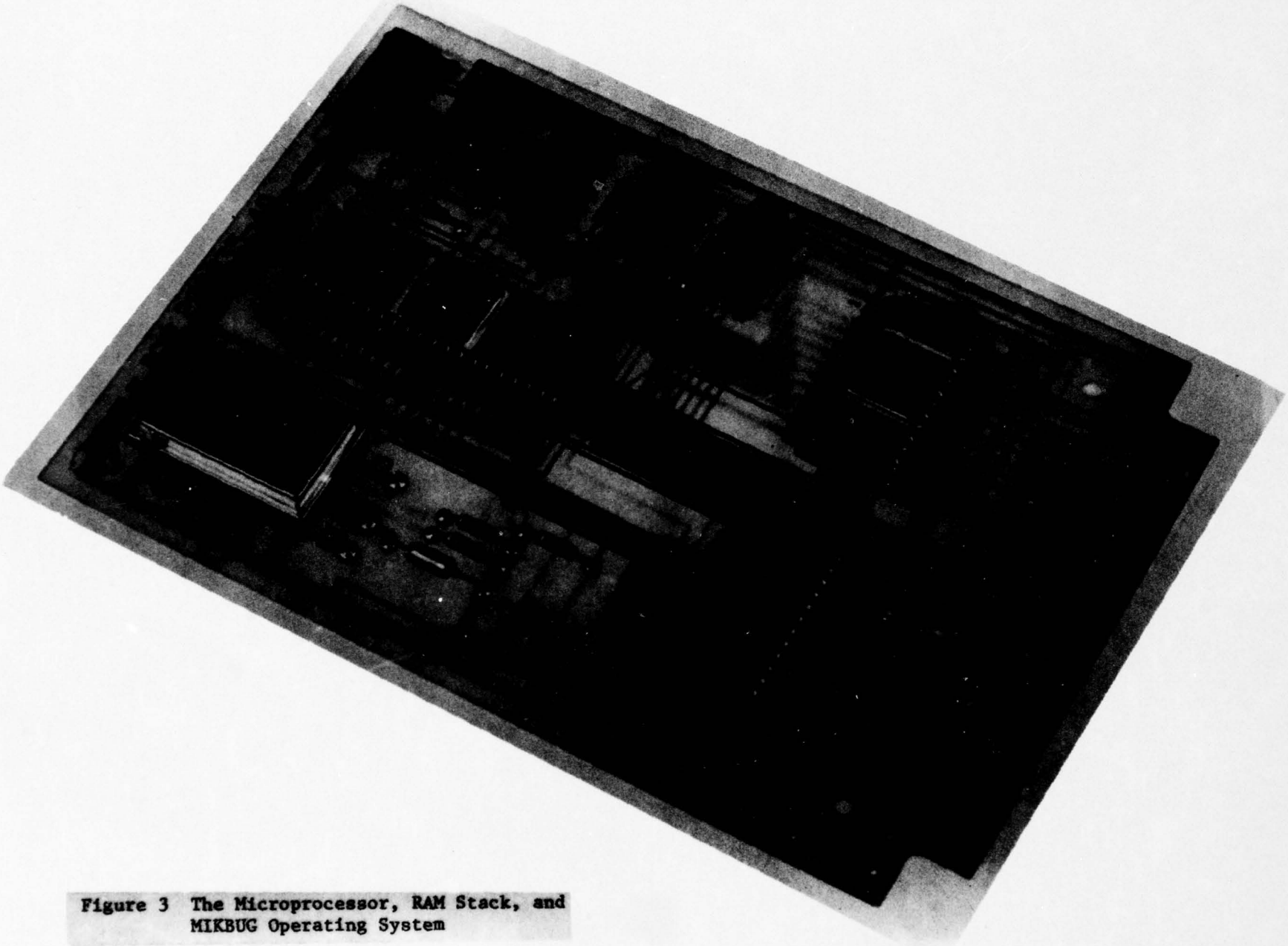


Figure 3 The Microprocessor, RAM Stack, and MIKBUG Operating System

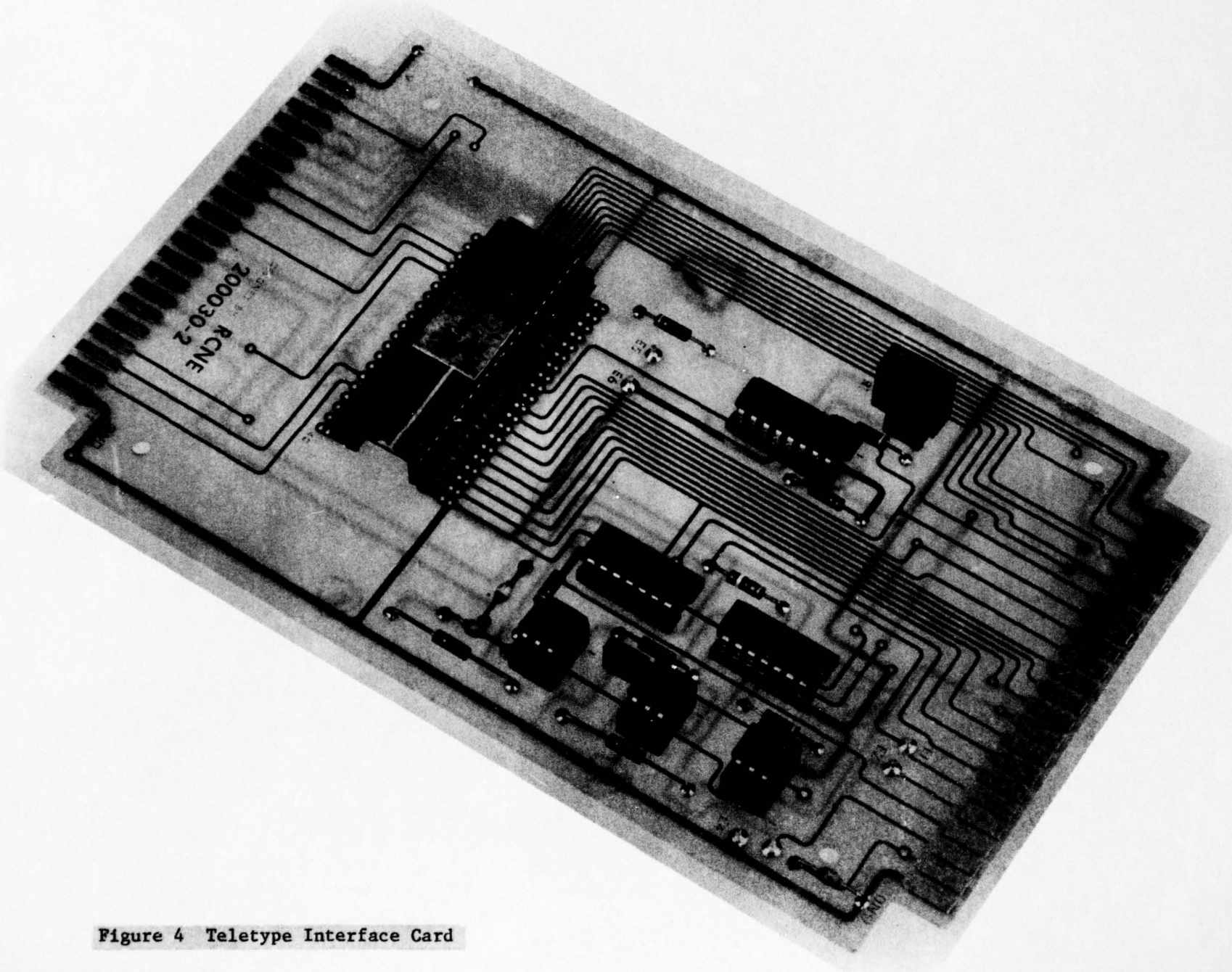


Figure 4 Teletype Interface Card

a variety of band rate data transfers. The drawing number for this circuit board is 18876-200030. The output connection to the teletype is made on a second 44-pin connecter (located on the right side of Figure 4).

A general interface circuit board for interfacing control circuits was developed as shown in Figure 5. The address and data lines are brought onto the board. Additional space is provided for custom wiring as the interface situation would dictate. The I/O connections are made on a second 44-pin connector. The peripheral interface adapter integrated circuit is capable of providing 16 inputs or outputs or combinations of these with 4 control input-output lines capable of interrupting the microprocessor. More detailed specifications on the integrated circuits can be found in the "Systems Reference and Data Sheets," [2]. The drawing number for this board is 18876-200036.

The microprocessor operation is directly rlated to memory - both in the ROM and RAM. To support this effort two printed circuit boards were developed. The Read Only Memory, ROM, board is shown in Figure 6 and is capable of holding 1K x 8 bit bytes programmable memory. The INTEL 1702A integrated circuit was found suitable for this purpose. This circuit board contains 4 1702A chips and space for the addition of NAND gates for chip selection and decoding. The drawing number for this board is 18876-200032. The Random Access Board RAM, is shown in Figure 7. The corresponding drawing and diagram is 18876-200017. The RAM memory card contains 1K by 8 bit bytes. In addition, the addressing is in the form of a card select in 1K blocks. Additional memory is added simply by selecting another memory card. During software development, a considerable amount of RAM may be required to execute the program. As subroutines become finalized RAM is replaced with ROM in 1K bytes.

PRESENT COMPUTER ARCHITECTURE

Figure 8 shows the memory map for the current system. The system concept is for modifiable memory (read-write random access memory, RAM), to start at address 0000 and expand upwards in address as needed in the application. Conversely, the control system which is in some form of permanent memory (PROM or ROM), should be placed towards

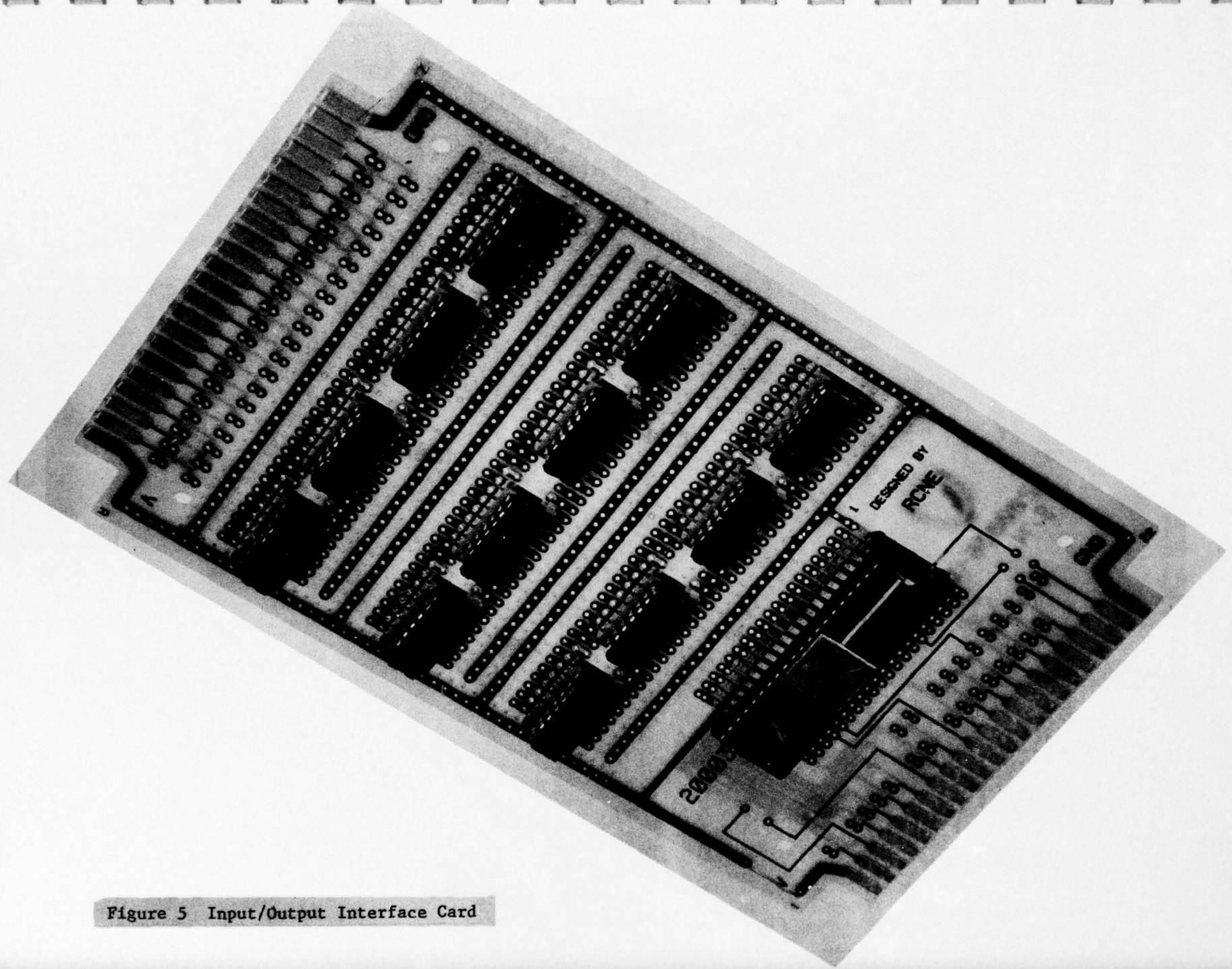


Figure 5 Input/Output Interface Card

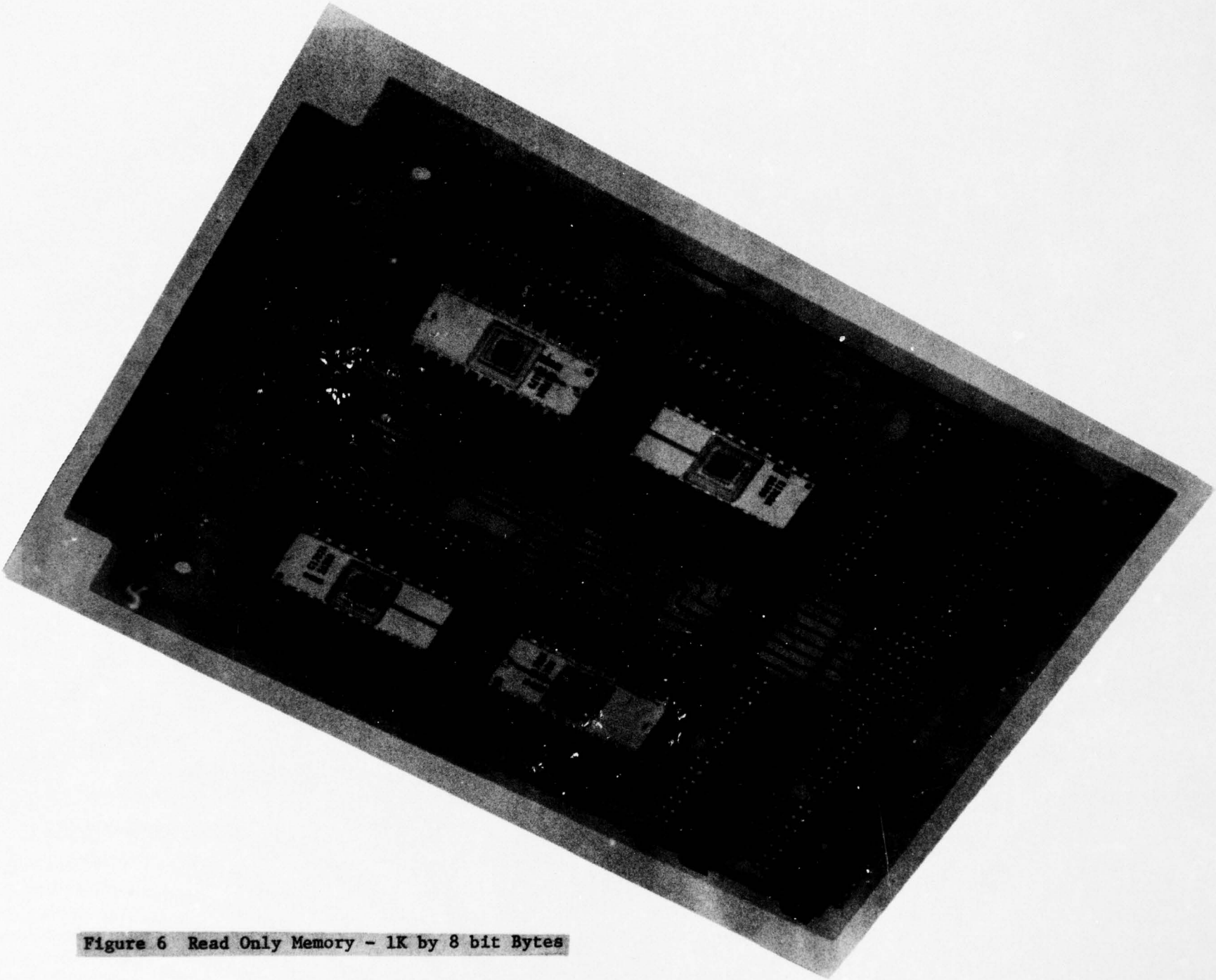


Figure 6 Read Only Memory - 1K by 8 bit Bytes

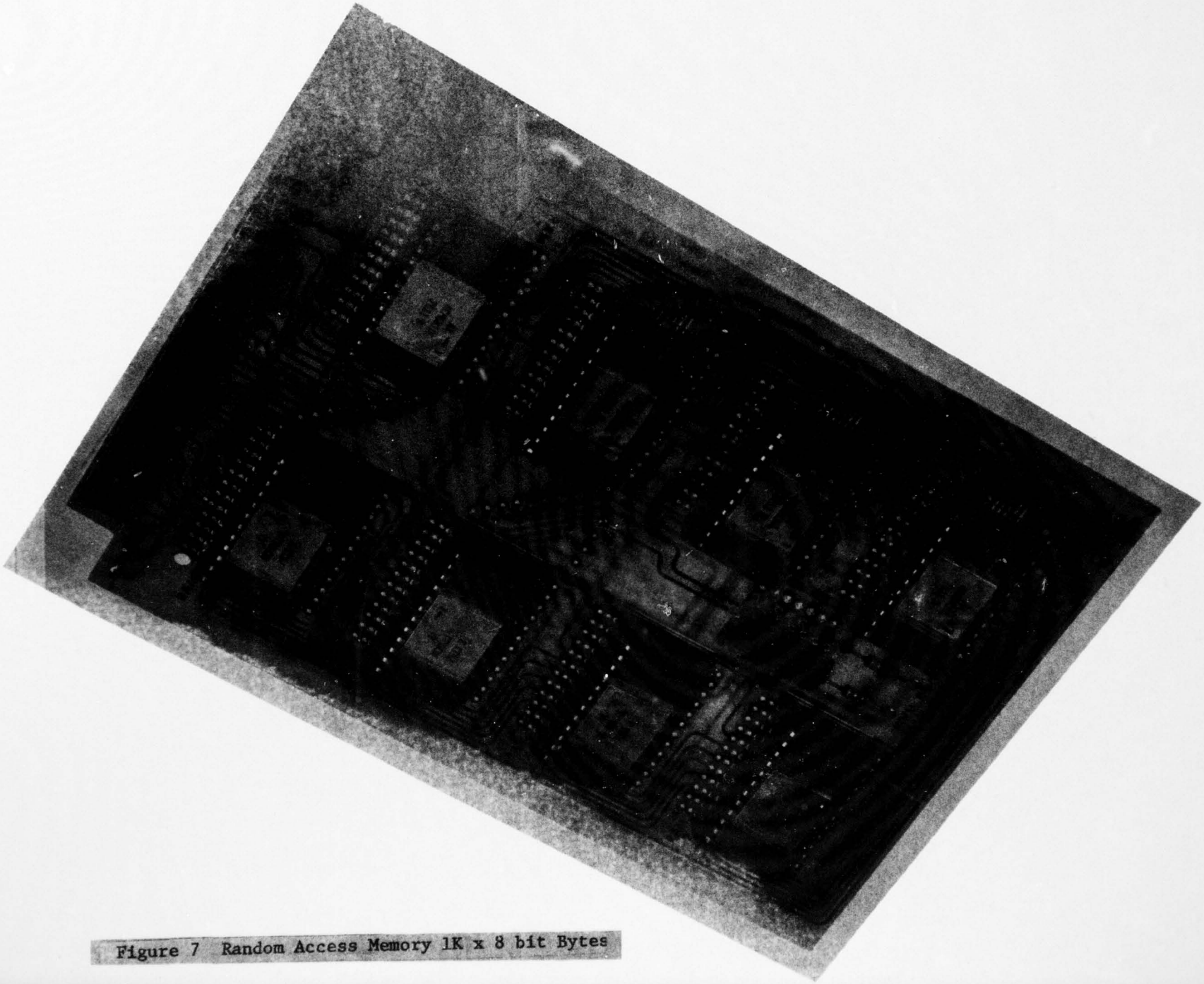


Figure 7 Random Access Memory 1K x 8 bit Bytes

MEMORY MAP

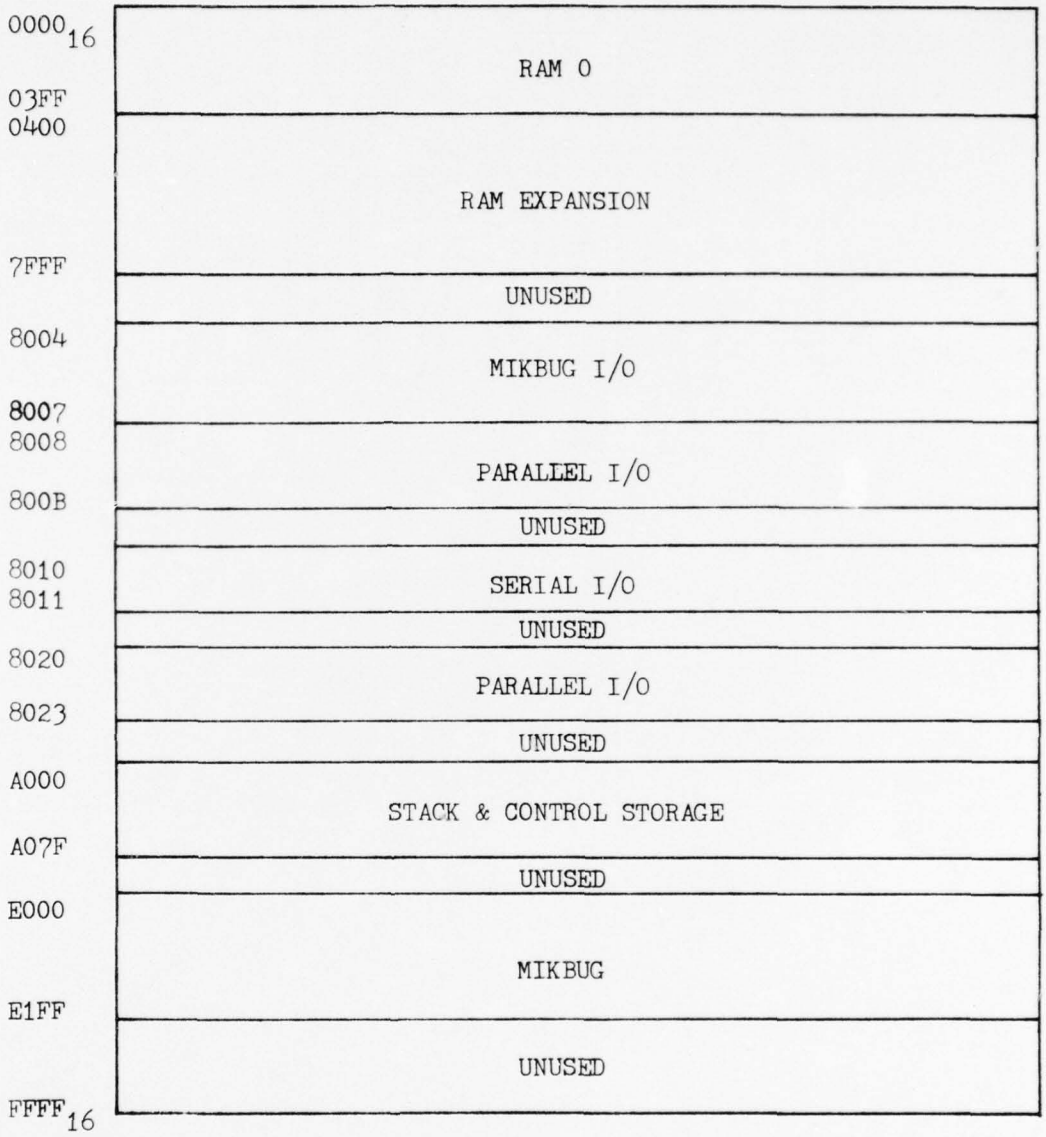


Figure 8

the top of memory FFFF and expand downwards. This concept is slightly modified when using the Motorola software product MIKBUG but when the actual operating system is in use the concept will be more accurately instigated. The input/output (I/O) devices are placed in memory starting at address 8000 and extending upwards in memory as needed. The MIKBUG operating system requires the parallel I/O device located at 8004 and all other I/O devices shown are application oriented. The system boards which are used in constructing this system and described elsewhere in this report were designed around this concept. A new operating system, GBUG, described in this report also follows this design.

SUPPORT CAPABILITY WHEN USING MOTOROLA MIKBUG OPERATING SYSTEM

The basic functions of the Motorola product, MIKBUG, are documented in their Engineering Report 100 "MCM6830L7 MIKBUG/MINIBUG ROM". Summarizing, MIKBUG takes care of interrupt handling and has the minimal necessary commands for program debug on the system. It has commands which allow for loading and punching a paper tape, examining and exchanging memory, examination of registers and going to the user program. These commands, while enough for operation, limit the program developer and require a great deal of time spent in familiarization before effective use of these commands is possible. For these reasons, a new operating system GBUG has been developed and will be discussed later in this report.

Besides the basic functions described above and documented in the Motorola literature, MIKBUG has several other useful support routines. In the following paragraphs, these will be discussed with the assembly language commands necessary to use the routines.

The first of these routines outputs a "?" to the system teletypewriter and then proceeds to the MIKBUG control loop. This routine can be used to handle any illegal condition in a user program. To use this routine the following code should be inserted in the program at the point where one wishes to go to the routine:

```
JMP $E040.
```

The dollar sign specifies that the arithmatic base is sixteen.

Another useful routine reads four hexadecimal numbers typed on the system teletypewriter and places them into the microprocessor's index register. At the point in code where one wishes to read the information, place the instruction:

```
JSR $E047.
```

The next instructions can assume the read values are in the index register. This routine destroys the previous contents of the two accumulators as well as the index register. Provision may be made to save and restore the accumulators' previous contents if desired. If the characters entered are not hexadecimal (not 0-9 or A-F) then control is transferred to the MIKBUG command loop.

To input a byte (2 hexadecimal characters) into accumulator A, one should enter the following code:

```
JSR $E055.
```

The next instruction may assume the accumulator now has the inputted value. As with the previous routine, entry of illegal characters causes program control to be transferred to MIKBUG. The routine destroys the contents of the B accumulator as well as the A accumulator's contents.

Next, to output a byte to the control teletypewriter, one may point the index register to the byte and jump to a MIKBUG routine. The following code accomplishes this:

```
LDX  Address of Byte to be output
JSR $E0BF.
```

The addressing mode of the load index register command will usually be "immediate". The routine does not affect the B accumulator, destroys the contents of the A accumulator and causes the index register to contain the address entered plus one when control is returned to the program using the routine.

A routine to output one byte in hexadecimal followed by a space on the control teletypewriter can be entered by the following routine:

```
LDX  Address of Byte
JSR $E0CA.
```

The registers are affected in the same manner as was described in the preceding paragraph. An extension of this routine is to print two sequential bytes from memory followed by a space. This is accomplished by the following code:

```
LDX  Address of first byte
JSR $E0C8.
```

The index register will contain the address placed in it plus two and all other registers will be as previously described.

To output a space to the control teletypewriter, insert the following sequence into the program:

```
JSR $E0CC.
```

Only the contents of the A accumulator are destroyed.

The previous routines do the conversion from the American Standard Code for Information Interchange (ASCII) to the hexadecimal language of the microprocessor to allow for ease of number entry. Sometimes though, one wishes to input and output characters in ASCII. To facilitate this one may use:

```
JSR $E1AC
```

to input an ASCII character into the A accumulator. No other registers are disturbed. To output an ASCII character from the microcomputer the user may utilize the following code:

```
LDA A Character
JSR $E1D1.
```

The addressing mode of the load the A accumulator will almost always be "immediate". To output a string of ASCII characters from memory to the control teletypewriter one may use the following:

```
LDX  start address of message
JSR $E07E.
```

The last character in the string must be the EOT (end of transmission) signal or 04 in the hexadecimal.

Once one has assembled the microcomputer and become familiar with the control system's commands there is a need for a simple program to exercise the computer and the I/O hardware. The following programs do just this. The first program inputs a string of ASCII characters to memory starting at the address contained in memory locations A070 and A071. The second program outputs the message from memory to the teletypewriter. Both programs are written in subroutine form to allow for use by many programs. First, the Read program:

```
READ    LDX      $A070    START OF TEXT IN MEMORY HERE
LOOP    JSR      $E1AC    READ ONE ASCII CHARACTER ROUTINE
        STA A    X        STORE READ CHAR IN MEMORY POINTED BY X
        INX               INCREMENT X REGISTER
        CMP A    #$04     WAS EOT ENTERED?
        BNE      LOOP     BRANCH FOR NEXT CHAR,NO EOT
        RTS               RETURN CALLING PROGRAM, EOT
```

The program takes 14 bytes in memory when assembled.

The Write program looks as follows:

```
WRITE   LDX     $A070     START OF TEXT
        JSR     $E07E     PRINT TEXT STRING
        RTS               RETURN
```

This program takes 7 bytes and could be shortened by one byte by replacing the JSR instruction with the JMP instruction and deleting the RTS. This can be done by allowing the calling routine to regain control after the MIKBUG routine has finished outputting the character string rather than waiting one more return from subroutine (RTS).

In order to communicate with the LCSS the microcomputer must convert from it's character set ASCII to that of the LCSS, Fieldata. This conversion is done by a table look up method. The LCSS also expects even parity so the microcomputer generates it also by the following program written in assembly language where the character in the A accumulator needs parity generated:

```
EPARITY   PSH   B          SAVE USERS B REGISTER
          ROL   A          IGNORE B7
          CLR   SCRATCH    ZERO SCRATCH MEMORY
          LDA   B #8       SET BIT COUNTER
PLOOP     ROL   A
          BNC   NCARY
          INC   SCRATCH    COUNT I'S
          DEC   B          COUNTER
          BNE   PLOOP      DO 8 BITS
          ROR   SCRATCH    ODD IF CARRY SET
          BNC   EVEN       EVEN PARITY ALREADY
          ADD   A #$80     MAKE EVEN
EVEN      PUL   B          RESTORE B
          RTS              RETURN TO CALLING PROGRAM
SCRATCH   RMB   1
```

An attempt to create a LCSS monitor program was made. Connections were made to the LCSS data bus and appropriate software was written. The monitor hardware worked successfully except that the LCSS paper tape reader was faster than the TTY I/O unit available to the microcomputer. This meant that more information was coming into the system than could be output due to the output device's speed. The microcomputer needs to have a parallel terminal for monitoring the LCSS capable of receiving approximately 600 bytes per second or a serial device capable of 9600 band. The microcomputer was proven to work correctly in this application by allowing it to store the backlog to queue it to the output device as

it became ready and by starting and stopping the LCSS tape reader to avoid overflowing the buffer. In this way, the microcomputer was shown to have sufficient speed to capture the data and do a Field data to ASCII conversion and the proper buffering. Upon acquisition of the proper output device the monitoring software will perform the monitor function. The monitor written assumes a serial device connected to the ACIA addressed at 8010-8011. To use a parallel monitor only the driver for the monitor need be changed (routine MONWRT). The routine assumes that the control console talks to MIKBUG.

MONITOR SOFTWARE

```
ACIACR   EQU              $8010
PIAIN    EQU              $8008
FLDASC   EQU              $0196
MONITR   LDX              #MONIRQ
         STX              $A000        SET UP IRQ
         LDX              #0           INITIALIZE X,B
         CLR              B
         CLR              PIANI+1      INITIALIZE
         CLR              PIAIN        LCSS INTERFACE
         LDA      A       #9           TTY MONITOR
                                       USE $11 FOR
                                       HIGH SPEED UNIT
         STA      A       ACIACR
         SEI
         LDA      A       #5
         STA      A       PIAIN+1      SET IRQ
         CLI                           ALLOW IRQ
MLOOP1   TST      B                    QUEUE EMPTY?
         BEQ      MLOOP1               YES
         LDA      A       0,X          NO,OUTPUT
         JSR      FLDASC               CONVERT
         JSR      MONWRT               OUTPUT
         SEI                           BUFFER HOUSEKEEP
         INX
         CPX      $0100
         BNE      CONT
         LDX      #0000                MAKE CIRCULAR
```

```
CONT        DEC     B                       ONE LESS IN QUEUE
            CLI
            BRA     MLOOP1
MONWRT      PSH     A
RLOOP       LDA     A       ACIACR          CHECK STATUS
            ASR     A
            ASR     A
            BCC     RLOOP                   NOT READY
            PUL     A                       READY
            STA     A       ACIACR+1        OUTPUT
            RTS
MONIRQ      LDA     A       PIAIN           GET DATA
            TSX                             GET X-REG
            LDX             3,X
            STX             SCRAT
            ADA     B       SCRAT+1         LSB'S
            STA     B       SCRAT+1         COMPUT NEXT ADDRESS
            LDX             SCRAT           IN QUEUE
            STA     A       0,X
            TSX                             GET SP
            INC             1,X             ONE MORE IN QUEUE
            BUS             MONBUP          QUEUE OVERFLOW
            RTI
MONBUP      LDX             #TEXT
MLOOP3      LDA     A       X
            CMP     A       #4              EOF?
            BEQ             TTYOUT          YES
            JSR             MONWRT          OUTPUT 1 CHAR
            INX
            BRA             MLOOP3
TTYOUT      LDX             #TEXT
            JSR             $E07E           PRINT USING MIKBUG
            JMP             $E0E3           MIKBUG CONTROL
TEXT        FCB             $D,$4,,,0
            FCC             /BUFFER CAPACITY EXCEEDED./
            FCB             04
            END
```

GBUG - A NEW OPERATING SYSTEM

As was mentioned previously, the Motorola operating system MIKBUG lacks several features. For this reason a new operating system, GBUG, was designed. In addition to the features provided by MIKBUG, GBUG has commands to accomplish a block transfer, display memory, punch an end-of-file symbol on the paper tape, reset the system, and rewind the paper tape drive. GBUG was specifically written for devices on location at UAH which are quite similar to the devices in use with the LCSS. The commands for GBUG are designed to be more easily explained to the inexperienced thus shortening the learning curve. The GBUG operating system is presently working in read write memory of the microcomputer system. In the future, GBUG along with other support software should be placed in read only memory.

GBUG is an attempt at modular coding. It was written so that every possible routine is available to the user for his program's use. In this way, routines such as the I/O (input/output) programs need not be rewritten from one test tape to another. GBUG in it's present form recognizes the four different logical devices of a computer system: the reader, the punch, the list, and the console. GBUG commands are only accepted from the console. In Version 1.0 of GBUG, there are two different assignments of actual devices to logical devices. The first assignment, which is the manner in which GBUG initializes is for each logical device to be the TTY connected to the Serial Interface (ACIA) at address 8010 and 8011. In other words, the initial state of GBUG utilizes the TTY as it's only I/O device. To command GBUG to the other assignment, the "A" command is used. (To be described in detail later.) In this alternate device assignment, the high speed paper tape reader is the reader, the TTY is the punch and list device and the console input is the TTY keyboard while the console output is a parallel monitor. Different versions of GBUG may be generated to allow for different assignments without any major changes in the overall software. Figures 9 and 10 show some of the I/O equipment used with GBUG 1.0.

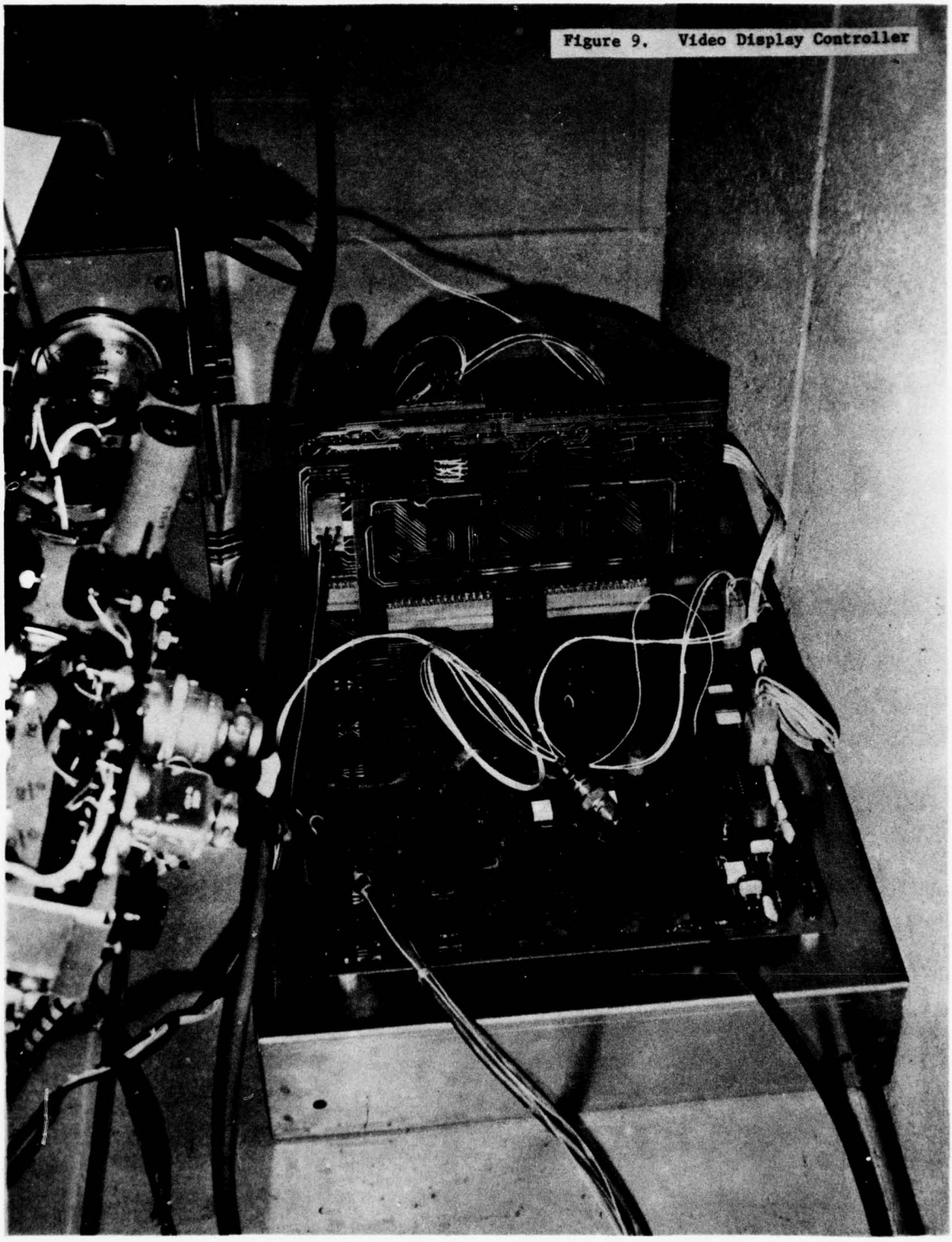


Figure 9. Video Display Controller

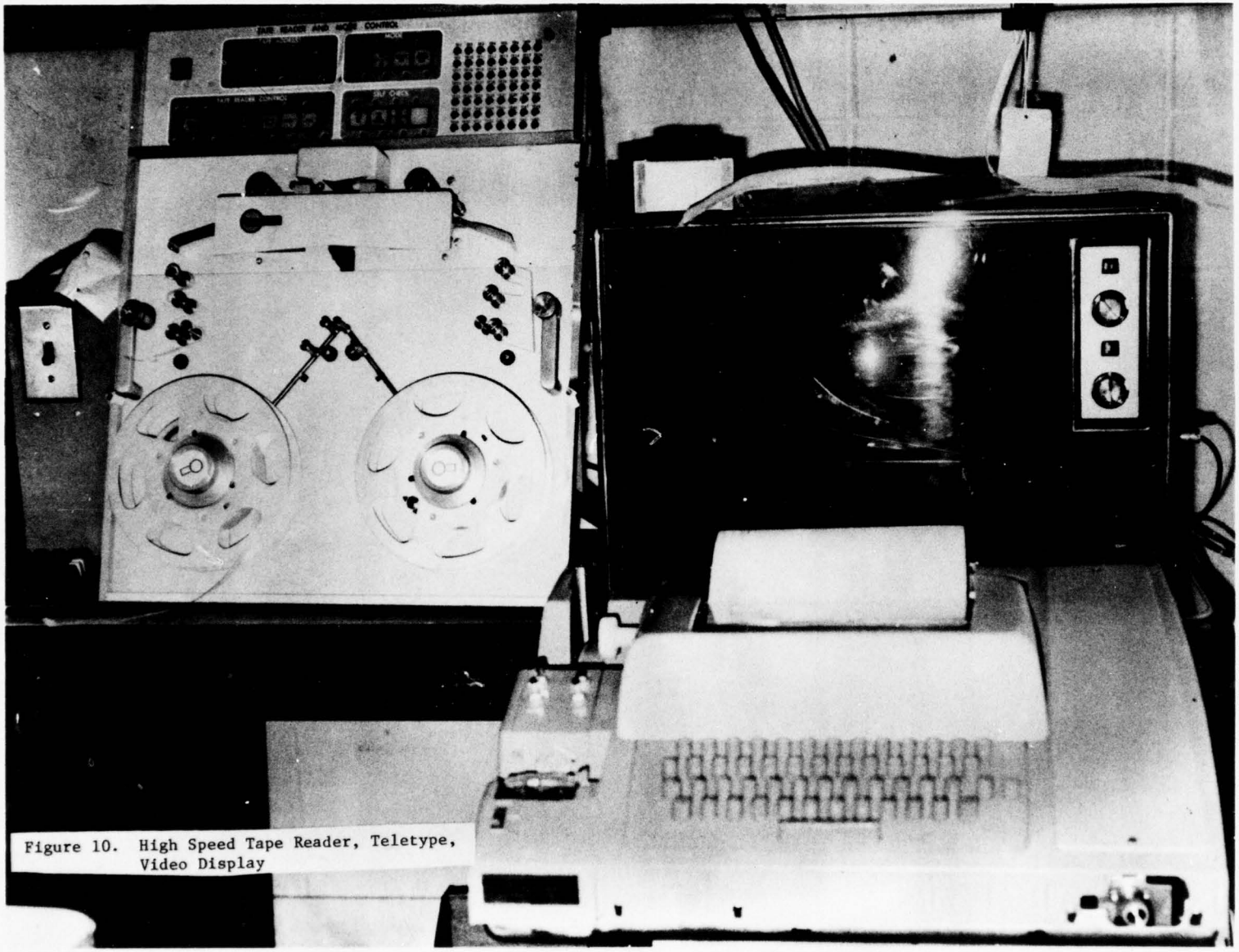


Figure 10. High Speed Tape Reader, Teletype, Video Display

The commands for GBUG are as follows: A, B, D, E, G, L, M, N, P, R, W, X and '.'.

The "A" command is simply given by typing A followed by a carriage return. This command assigns the alternate device set. If the command is repeated, the system is as if no command were given.

The "B" command instructs the computer to transfer a block of memory from one spot to another. It is of the form:

B<START>,<STOP>,<TO> ↵

where the ↵ denotes the carriage return. Each field can be thought to consist of four hexadecimal digits. Leading zero's need not be entered and if more than four digits are entered the last four are used. To illustrate this command, the following command transfers memory between location 0400 and 0800 (inclusive) to a block of memory starting at 0000:

B400,FF0800, ↵

Note that if 0 is the address to be used only the proper delimeter (, or ↵) need be entered. This command errs out if three arguments are not entered (E 04) or a non-hex character is entered in (E 02), except of course the ',' or ' ↵'.

The 'D' command displays the contents of memory in readable form. The form of the command is:

D<START>,<STOP> ↵.

If the second argument or its delimeter is not entered only one byte is displayed. For example,

D4 ↵

causes a print out of the address 0004 and it's contents. Likewise,

D4,00F ↵

causes a display of the address 0004 and the contents of memory locations

0004 through 000F. For readability reasons, the command formats 16 bytes to a line and prints the address of the byte which starts each line. If a non-hex character is entered an E02 condition exists and is so flagged.

The 'E' command punches an end of file mark and trailer on a paper tape which signals the end of a load for GBUG, MIKBUG or other Motorola software.

The 'G' command is used as a "GO TO" user program command. It has the form:

G<ADDRESS> ↵

The address is evaluated the same way as parameters for the block transfer command.

G100 ↵

causes program execution to begin at address 0100 with the registers having the contents of what ever they were prior to entering the operating system. On start up, their contents are undefined. To define or examine the registers, the 'X' command is used. If a non-hex character is entered, an E 02 signal flags the operator and outputs another prompting signal (>). Typing G ↵ causes program execution to begin at 0000.

To load a paper tape from the reader use the 'L' command. This command works exactly like the L command documented in Engineering Note 100 mentioned before except that the TTY if it is the reader does not print a copy of the tape when loading. The paper tape is checked by a checksum for each record and an error E 03 exists if a checksum error exists or an invalid character is read. If an error exists, the tape may be rewound to the start of the record and re-read. The whole tape need not be re-read.

The GBUG 'M' command is similar to the MIKBUG M command, however they have significant differences. The form of the GBUG command is:

M<ADDRESS> ↵ .

This causes the contents of the memory to be displayed followed by a dash. At this point, the operator has three choices: to change the contents, not change the contents and examine the next byte, or not change the contents and return to the command loop. To change the contents one types the two character representation of the memory the user wishes in that address. If a non-hex character is entered an E 02 is typed and control is returned to the command loop without modifying the memory. The write to memory is attempted and read to see if the correct contents were written, if unsuccessful an E 05 is typed and control returned to the command loop.

If the change was successful, the next address in memory contents are displayed where the options are again open to the operator. To merely examine the next byte while not affecting the contents of the present location, simply hit the space bar and the options are again open. To terminate the command without changing memory type in a carriage return. Control is then returned to the command loop.

To produce a leader, hit N on the console keyboard and turn on the punch. Sixty nulls are then output to the punch. This command will be terminated after sixty nulls or when another key is pressed. To produce longer leaders hit N repeatedly.

To punch a MIKBUG/GBUG compatible tape file use the P command. The arguments work the same as for the D command except two arguments must be specified or an E 04 condition exists. For tape formatting see Engineering Note 100 by Motorola.

To reinitialize, the user's stack pointer, and verify that the GBUG command loop is still in control, the R command may be used. The command causes the printing of the command program (GBUG 1.0) and a prompting symbol. A carriage return is not necessary for this command.

To rewind the paper tape reader (if the high speed unit is used), the W command is used.

As was mentioned before, the X command may be used to examine and modify the contents of the registers that will be returned to the user's program by the G or . commands. The X is followed by either an A, B, C, X, P, or S to designate the A accumulator B accumulator, condition codes, index register, program counter or stack pointer, respectively. If the register is 8 bits wide, (A, B, C) the user may change the contents by typing the two hex characters or not change the contents by typing a space or carriage return. When the register is 16 bits wide, the operator may change, not change or exit the first byte as with the M command and the second byte of the register has the same options as the 8 bit registers. If the stack pointer is changed it will cause all other registers to be changed as their location in memory during GBUG operation is relative to the user stack pointer. Therefore, in initializing all registers, the stack pointer should be loaded first. It is suggested that either A020 be used as the initial pointer address (as GBUG initializes it) or somewhere in the user RAM below to avoid conflict with the RAM that GBUG uses in its control loops. If an invalid register is specified or a non-hex change is attempted, an error, E 02, is produced.

To resume operations as per the contents of the registers (as opposed to the G command which alters the program counter use the . command. This enables the interrupt feature so the user program must disable this feature if it is not desired. He may also delay it's operation by use of the SEI (set interrupt mask) command in his code.

The verison of GBUG that is placed in ROM will handle the interrupts that are generated but in this verison GBUG is subservant to MIKBUG which has control of the interrupts. This was necessary for debug purposes.

When initialized, the GBUG operating system types '*GBUG 1.0*' on the console device followed by the printing of '>' as a prompter. The prompting symbol is printed whenever GBUG has finished the command

requested of it. GBUG allows the operator to interrupt current processing by typing any character on the console. The letter punched will be considered a GBUG command and will be executed if possible. Illegal commands will cause the E 01 flag to be printed on the console device. The interrupted processing may be returned to by the '.' command. The use of the interrupt feature applies to any GBUG command which does not expect an input from the physical device identified logically as the console or is a short closed routine. In other words, after the arguments (if any) are obtained, console interrupts are allowed on the G, N, W, and '.' commands. If the TTY is not the reader, then the L command also allows for an interrupt. If the user wishes to input data from the console, he should first disable the interrupt feature by jumping to the DISINT (disable interrupt) routine. To renable the feature, ENINT may be used. Documentation on these routines and other routines including I/O may be found in Appendix B along with the listing of GBUG 1.0.

RECOMMENDATIONS

The hardware designed for the prototype system is complete except for two features. First, data bus drivers need to be added to the system should more than the present number of devices be needed in the system. This would best be done on a prototype basis by the creation of a bus compatible extender card with resident drivers. This would allow for virtually unlimited system expansion. The second feature need not be added except for testing purposes. That is an error detection scheme for the internal microcomputer. This is not necessary in this level prototype model but would probably be necessary in the field to insure reliability.

Software recommendations for this system are primarily extensions of the work done. More routines need to be added to ROM as the commonly needed LCSS functions are identified. Additionally, current mass storage media which are microcomputer compatible (such as the floppy disk) would allow for storage of all LCSS programs with the ability to almost instantly recall them. This would require an extension of the current operating system to handle the I/O for the unit.

REFERENCES

[1] Application Note 100, Motorola Semiconductor Products, Box 20912, Phoenix, Arizona 85036.

[2] System Reference and Data Sheets, Ibid.

[3] M6800 Microprocessor Applications Manual, Ibid.

[4] M6800 Microprocessor Programming Manual, Ibid.

APPENDIX A

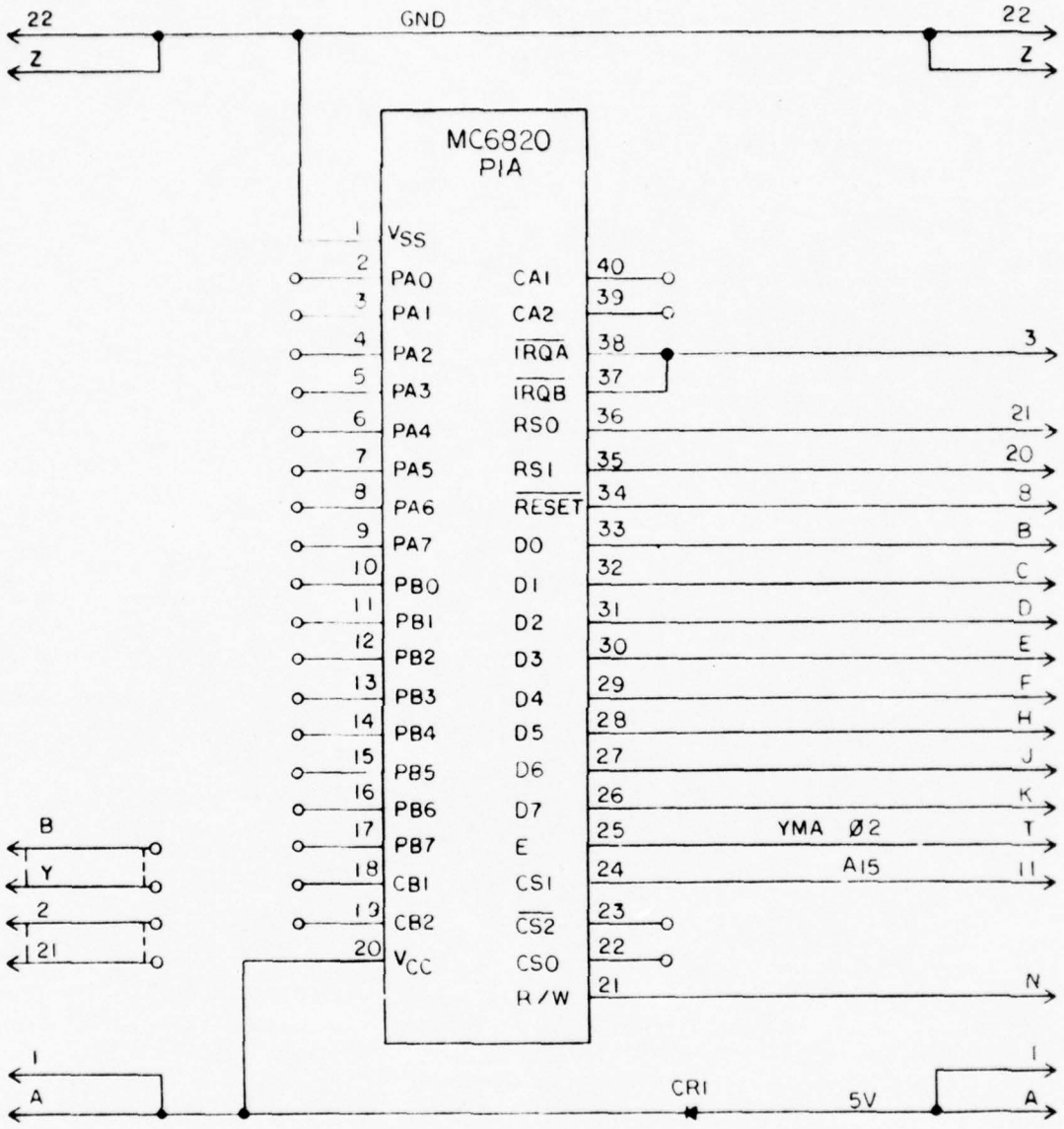


Figure A-1. Peripheral interface adapter universal board.

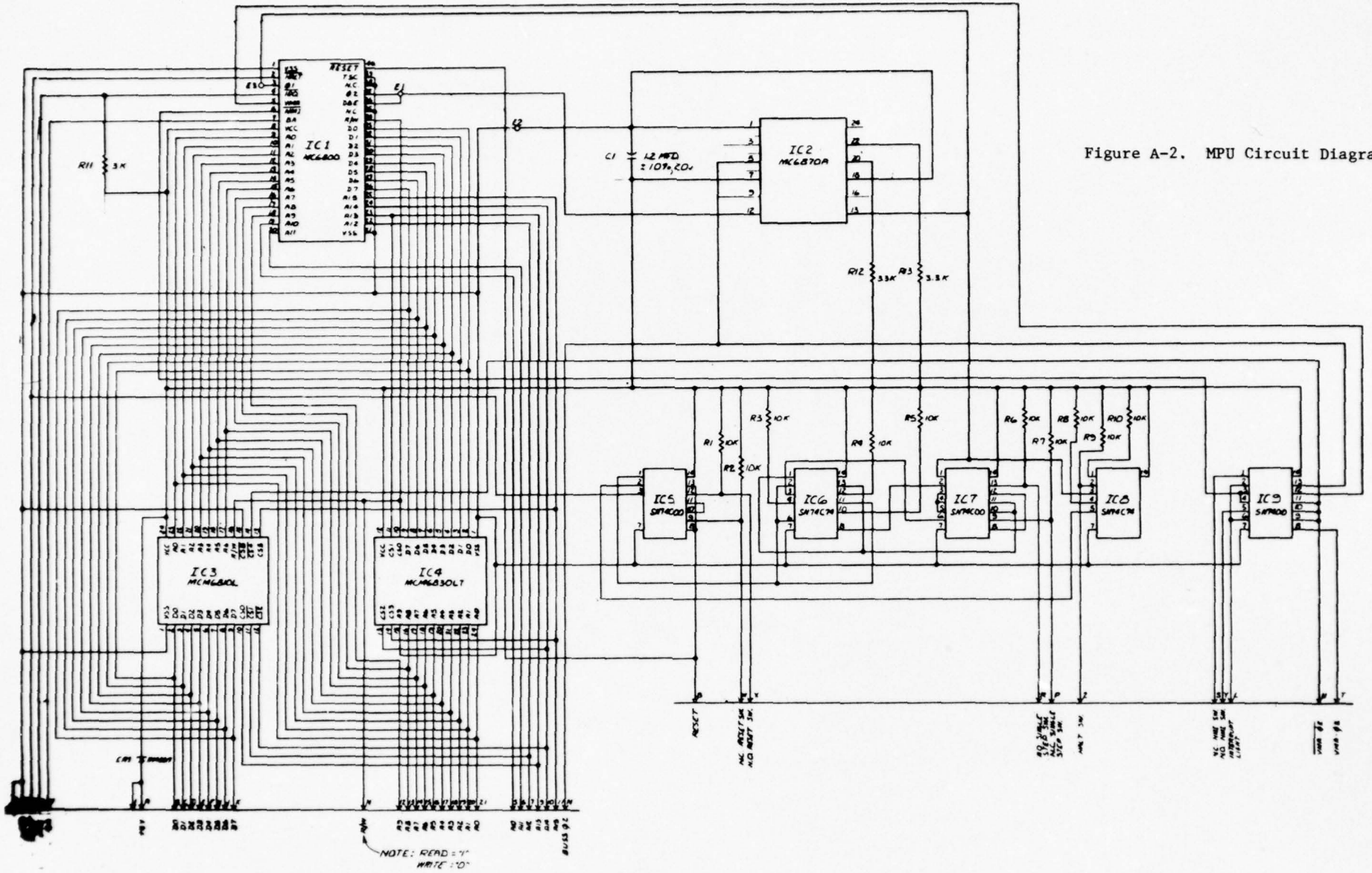


Figure A-2. MPU Circuit Diagram

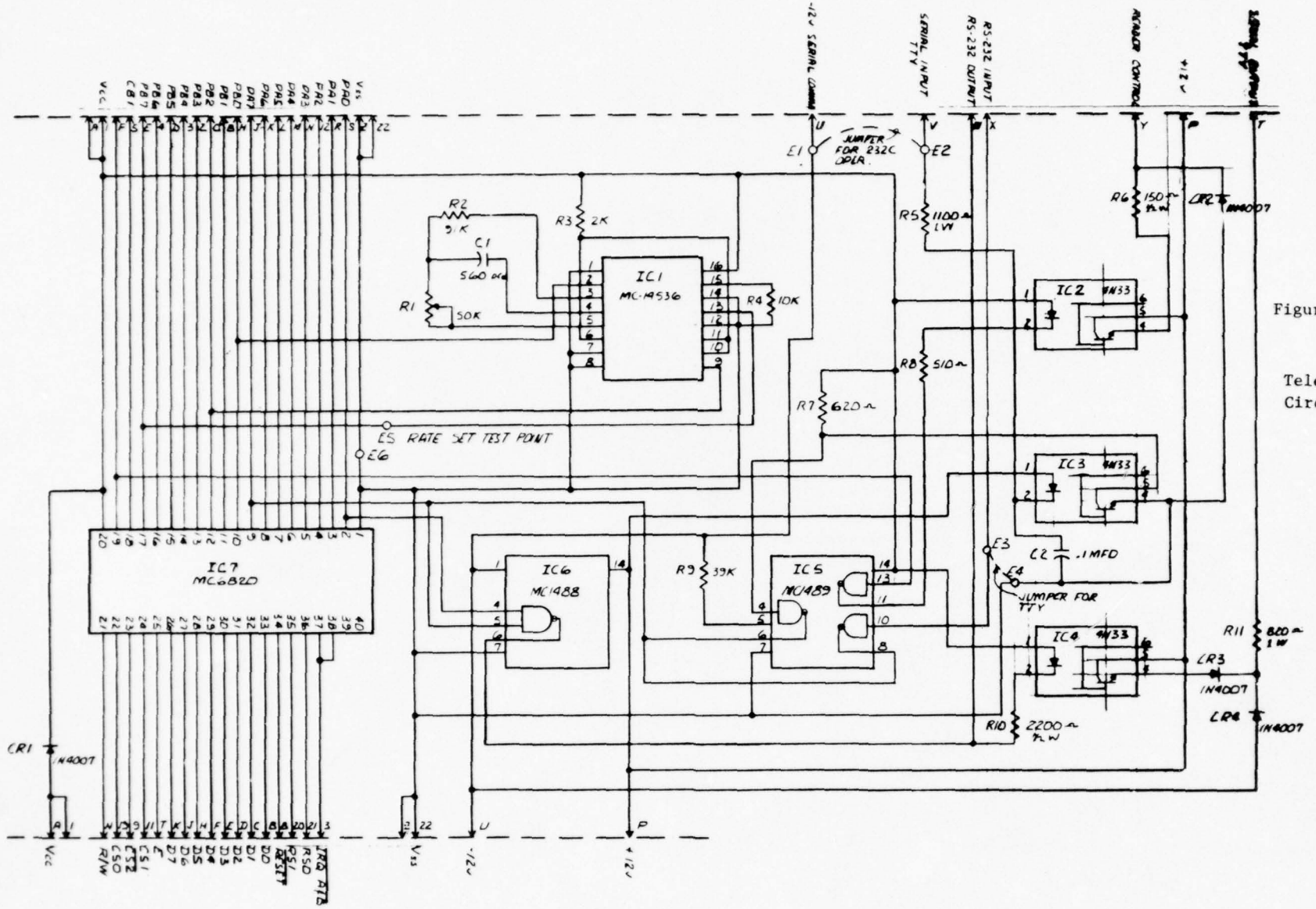


Figure A-3.

Teletype Interface Circuit Diagram

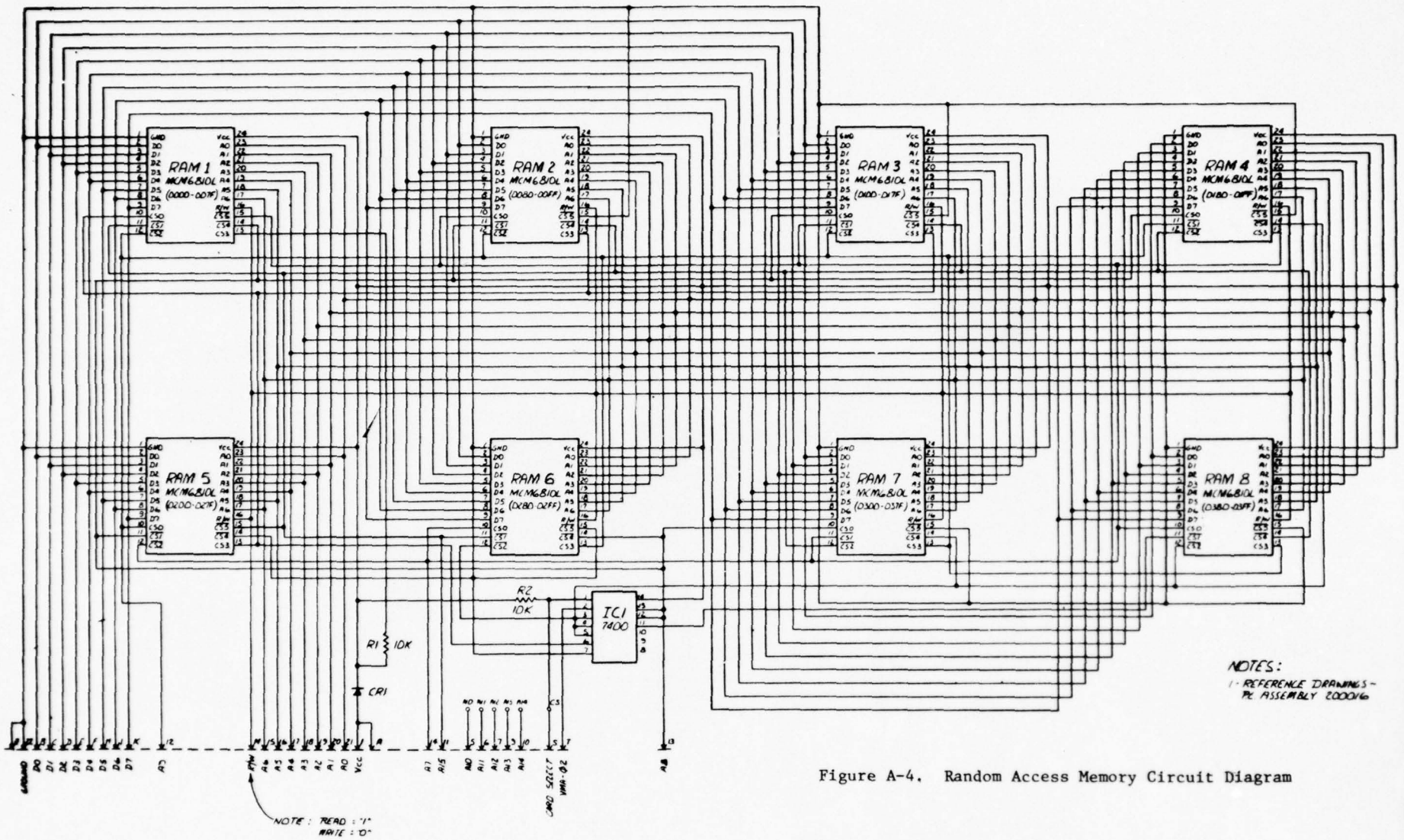


Figure A-4. Random Access Memory Circuit Diagram

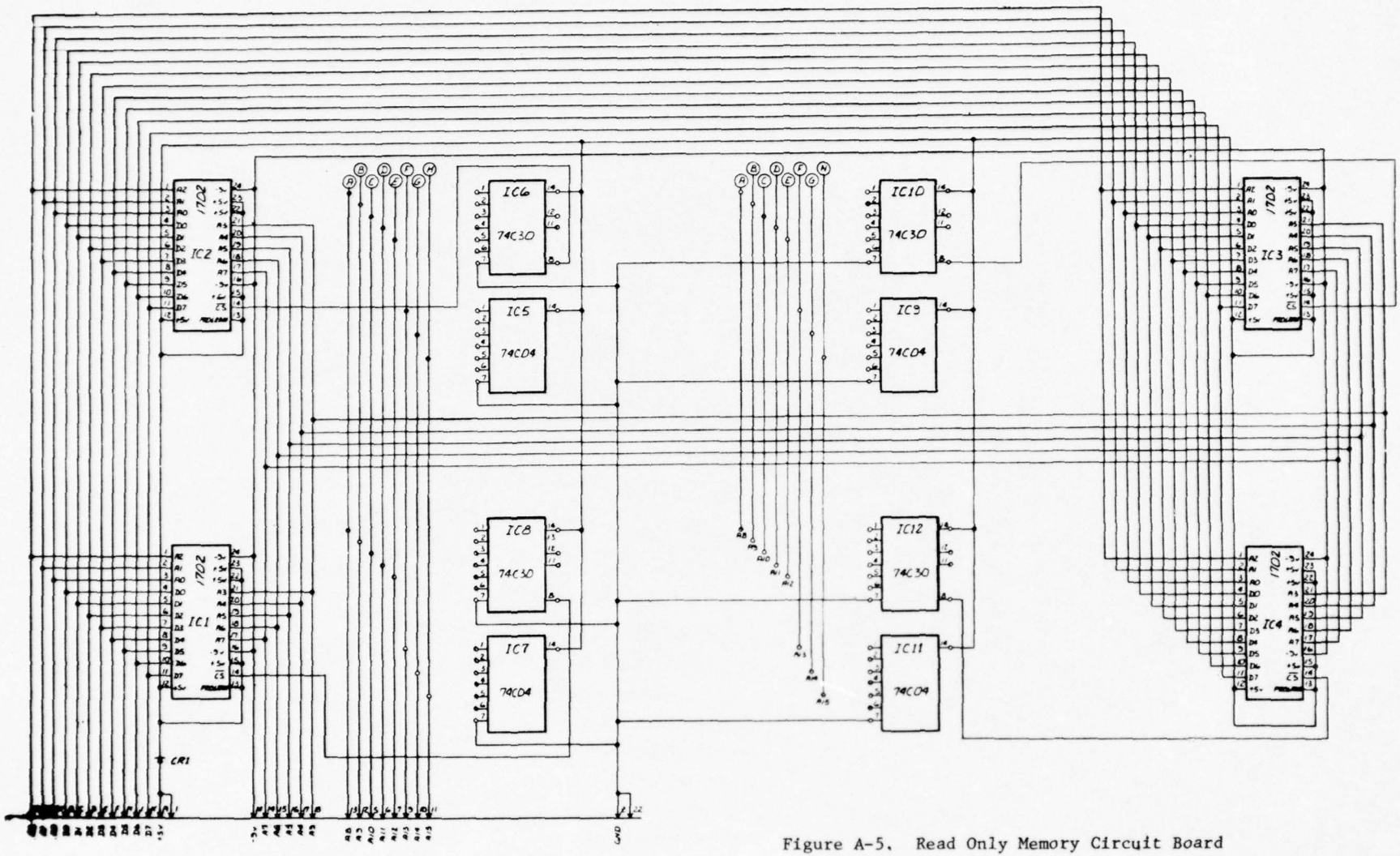


Figure A-5. Read Only Memory Circuit Board

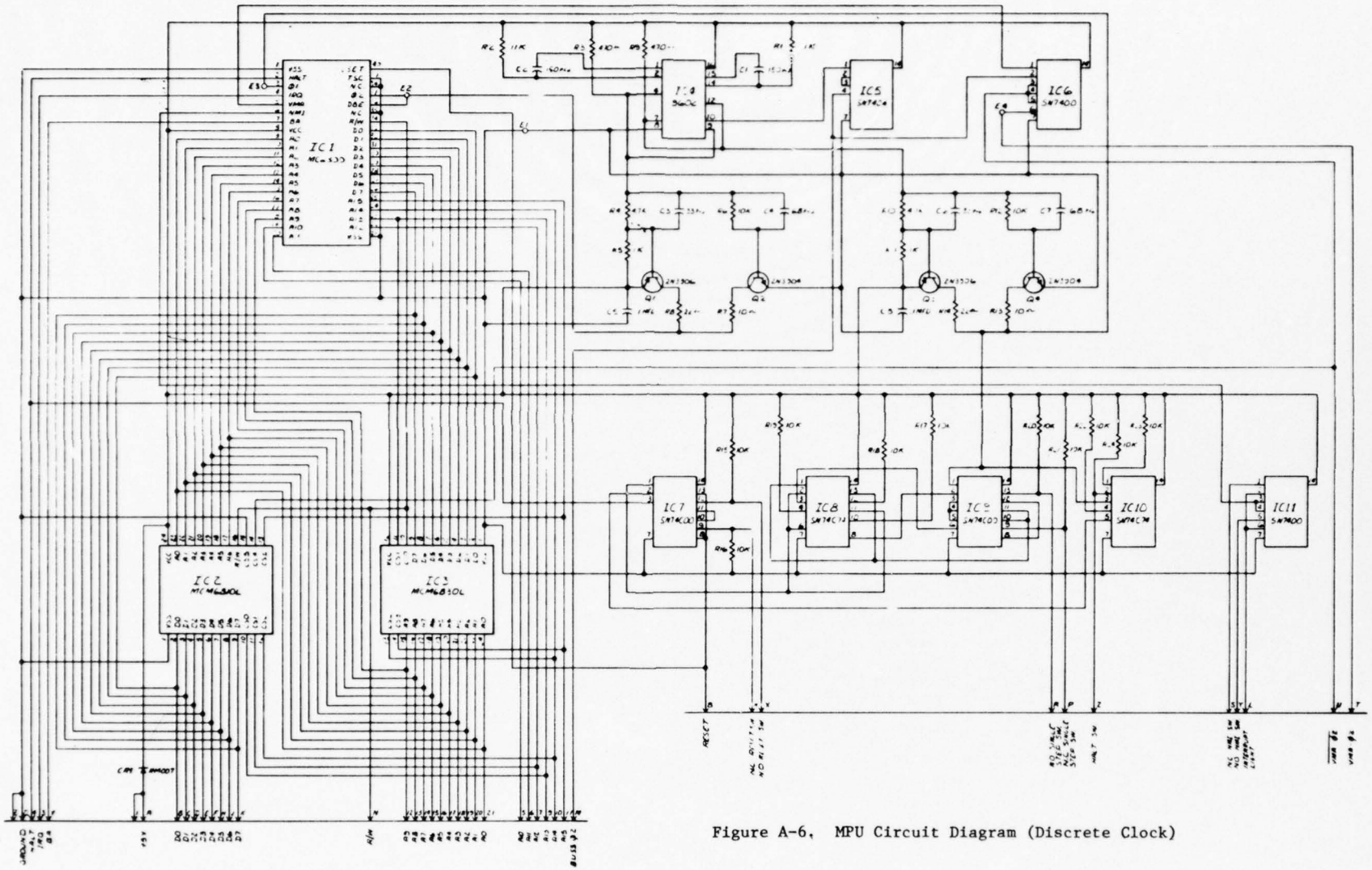


Figure A-6. MPU Circuit Diagram (Discrete Clock)

APPENDIX B

MOTOROLA M6800 CROSS ASSEMBLER, RELEASE 1.2

```
00001                    NAM     GBUG

00003                ************************************
00004                *                                  *
00005                *     GREENBUG  VERSION 1.0        *
00006                *                                  *
00007                *     A MC6800 OPERATING SYSTEM    *
00008                *                                  *
00009                *     AUTHOR: DAVID G. GREEN       *
00010                *                                  *
00011                *     DATE AUGUST 9,1976           *
00012                *                                  *
00013                ************************************

00015                ***VALID COMMANDS***

00017                *A    ASSIGN ALTERNATE DEVICE

00019                *B <START FROM>,<STOP FROM>,<START TO>   BLOCK TRANS

00021                *D  <START>,<STOP>   DISPLAY MEMORY

00023                *L    PUNCH EOF (S9)

00025                *G  <ADDRESS> ENABLE CONSOLE IRQ AND GO TO ADDRESS

00027                *L    LOAD TAPE FROM READER

00029                *M  <ADDRESS> MEMORY EXAMINE

00031                *N    PUNCH NULLS
                     *P    PRINT/PUNCH MIKBUG Compatible TAPE
00033                *R    RESET SYSTEM (NOT I/O)

00035                *W    REWIND HIGH SPEED PAPER TAPE READER

00037                *X  <REGISTER>   EXAMINE REGISTER

00039                *.    RETURN CONTROL TO USER PROGRAM AT POINT OF INT
00040 00B0                 ORG     $B0
```

```
00042                    ***POLLI1 - IRQ VECTORS TO THIS POINT.
00043                    *       CONSOLE DEVICE CHECK FOR INTERRUPT.
00044                    *       IF CONSOLE DEVICE DID IRQ
00045                    *       THEN JUMP TO COMMAND LOOP.
00046                    *       IF NOT CONSOLE THEN JUMP TO USER
00047                    *       IRQ HANDLER INDICATED BY [USEINT]
00048                    *       TTY KEYBOARD IS CONSOLE KEYBOARD
00049                    *       INPUT   N/A
00050                    *       OUTPUT   N/A
00051                    *       REGISTERS AFFECTED   A,CC
00052                    *           NOTE ALL USER REGISTERS ON STACK.
00053                    *       CLOSED SUBROUTINE?   NO, OPEN INTERRUPT ROUTIN
00054                    *       SUPPORT ROUTINES   GBUG

00056 00B0 36            POLLI1 PSH A
00057 00B1 B6 8010              LDA A   TTYCON    TTY IRQ?
00058 00B4 46                   ROR A
00059 00B5 24 04                BCC     INTOT1    BRANCH IF NOT
00060 00B7 32                   PUL A
00061 00B8 7E 010C CONINT       JMP     TRICK1    GO TO GBUG
00062 00BB 86 7E   INTOT1       LDA A   #$7E      TTY ENTRY
00063 00BD B7 A051              STA A   USEINT-1   STORE 'JMP'
00064 00C0 32                   PUL A
00065 00C1 7E A051              JMP     USEINT-1   DOUBLE JUMP
```

```
00068                    * INITIALIZATION POINT *

00070 00C4 7F A050 GB0G  CLR     DEVICE   ALL LOGICAL DEVICES ARE TTY

00072                    * DEVICE NAMES AND INITIALIZATION

00074        8008        PIVTRD  EQU     8008H    PIA CONNECTED TO TVT (READ)
00075        800A        PIVTWR  EQU     PIVTRD+2   PIA CONNECTED TO TVT (WRITE
00076        8009        PIVTC2  EQU     PIVTRD+3   WRITE CONTROL
00077        8010        TTYCON  EQU     $8010    ACIA CONTROL (TTY)
00078        8011        TTYIOL  EQU     TTYCON+1   I/O LINE OF TTY
00079        8020        PIARD   EQU     $8020    TAPE READ PORT
00080        8022        PIACON  EQU     PIARD+2    READ CONTROL
00081        0091        TTYIE   EQU     91H      INTERRUPT ENABLED (TTY READ)
00082        0011        TTYID   EQU     11H      INTERRUPT DISABLED (TTY READ)
00083        0060        STARTR  EQU     60H      FORWARD ON PTR
00084        0040        RWDRTR  EQU     40H      REVERSE ON PTR
00085        0000        STPTR   EQU     0        STOP ON PTR
00086        003E        SYSSYM  EQU     $3E      >

00088                    *      STORE VECTOR INFORMATION

00090 00C7 CE 02EA       LDX     #NMIHAN    NMI HANDLER
00091 00CA FF A006       STX     $A006    MIKBUG NMI
00092 00CD CE 00B0       LDX     #PCELL1    IRQ 1 HANDLER
00093 00D0 FF A000       STX     $A000    MIKBUG

00095 00D3 CE 8008       LDX     #PIVTRD    PARALLEL MONITOR
00096 00D6 6F 03         CLR     3,X
00097 00D8 6F 02         CLR     2,X
00098 00DA 63 02         COM     2,X      ALL OUTPUTS
00099 00DC 86 26         LDA A   #$26     OUTPUT CONTROL
00100 00DE A7 03         STA A   3,X
00101 00E0 6F 02         CLR     2,X      DUMMY WRITE ON CONSOLE
00102 00E2 CE 8020       LDX     #PIARD   PAPER TAPE READER
00103 00E5 86 06         LDA A   #6
00104 00E7 C6 60         LDA B   #60H
00105 00E9 6F 01         CLR     1,X
00106 00EB 6F 00         CLR     0,X
00107 00ED 6F 03         CLR     3,X
00108 00EF 6F 02         CLR     2,X
00109 00F1 A7 01         STA A   1,X
00110 00F3 E7 02         STA B   2,X
00111 00F5 A7 03         STA A   3,X
00112 00F7 86 03         LDA A   #3
00113 00F9 B7 8010       STA A   TTYCON
00114 00FC 86 11         LDA A   #ITYID
00115 00FE B7 8010       STA A   TTYCON

00117                    *  ENTRY POINT FOR ABORTED JOBS LOGICAL ASSIGNMENT
00118                    *     UNCHANGED AS WELL AS SYSTEM PERIPHERALS.
```

```
00120 0101 8E A030 ABORT  LDS   #STACK   USER'S SP
00121 0104 CE 057B        LDX   #GDMSG   TITLE
00122 0107 4F             CLR A
00123 0108 4A             DEC A
00124 0109 BD 03A2        JSR   OPSTNG   PRINT STRING ON CONSOLE
00125 010C 8F A008 TRICK1 STS   TSTACK   STORE USER'S SP
00126 010F 8E A042 SYSABT LDS   #SYSSP0  INITIALIZE STACK (SYSTEM)
00127 0112 7F 8022        CLR   PIACON   STOP PTR
00128 0115 8D 02          BSR   TRICK2   GET RENTRY POINT ON STACK
00129 0117 20 F6   RENTRY BRA   SYSABT
00130 0119 8D 024E TRICK2 JSR   DISINT   CLEAR INT
00131 011C 4F             CLR A
00132 011D 4A             DEC A
00133 011E CE 058A        LDX   #SYSTNG
00134 0121 8D 03A2        JSR   OPSTNG   PRINT PROMPTER
00135 0124 8D 02A9        JSR   RDCON    READ CHAR FROM CONSOLE
00136 0127 8D 0344        JSR   OUTCON   ECHO
00137 012A C1 41          CMP B #'A
00138 012C 26 04          BNE   GBN01
00139 012E 73 A050        COM   DEVICE
00140 0131 39             RTS
00141 0132 C1 45   GBN01  CMP B #'E
00142 0134 26 03          BNE   GBN02
00143 0136 7E 0561        JMP   MLOF
00144 0139 C1 44   GBN02  CMP B #'D
00145 013B 26 03          BNE   GBN03
00146 013D 7E 0449        JMP   DISPLY
00147 0140 C1 42   GBN03  CMP B #'B
00148 0142 26 03          BNE   GBN04
00149 0144 7E 0198        JMP   BTRANS
00150 0147 C1 4E   GBN04  CMP B #'N
00151 0149 26 06          BNE   GBN05
00152 014B CE 036A        LDX   #NOLLS
00153 014E 7E 0256        JMP   ENINT
00154 0151 C1 4C   GBN05  CMP B #'L
00155 0153 26 03          BNE   GBN06
00156 0155 7E 02DC        JMP   LOADER
00157 0158 C1 4D   GBN06  CMP B #'M
00158 015A 26 03          BNE   GBN07
00159 015C 7E 04B6        JMP   MEXCHG
00160 015F C1 50   GBN07  CMP B #'P
00161 0161 26 03          BNE   GBN08
00162 0163 7E 03D6        JMP   PRINT
00163 0166 C1 58   GBN08  CMP B #'X
00164 0168 26 03          BNE   GBN09
00165 016A 7E 052B        JMP   RECXNG
00166 016D C1 52   GBN09  CMP B #'R
00167 016F 26 03          BNE   GBN010
00168 0171 7E 0101        JMP   ABORT
00169 0174 C1 57   GBN010 CMP B #'W    REWIND TAPE
00170 0176 26 0D          BNE   GBN011
00171 0178 CE 0183        LDX   #STAY
```

```
00172 0178 86 40          LDA A  #$40.ORTR
00173 017A B7 8022        STA A  PIACON
00174 017D 7E 0256        JMP    EN.INT
00175 0180 20 FE    STAY  BRA    STAY       WAIT FOR INTERRUPT
00176 0182 C1 2E    GBN01 CMP B  #'.        RETURN PROGRAM AS IN REGISTER
00177 0184 26 03          BNE    GBN012
00178 0186 7E 04AC        JMP    GOTOD      ENABLE CONSOLE IRQ AND RTI
00179 0189 C1 47    GBN012 CMP B #'G
00180 018B 26 03          BNE    ERR1
00181 018D 7E 0494        JMP    GOTO
00182 0190 86 01    ERR1  LDA A  #1
00183 0192 7E 0260        JMP    SYSERR
```

```
00185 0196 C6 03    BTRANS LDA B  #3          ORDER: START,STOP,WHERE
00186 0198 17              TBA
00187 0199 CE A061         LDX    #BTINFO     PARAMETER STORAGE
00188 019E 8D 24           BSR    PARSCN      ACQUIRE PARAMETERS
00189 01A0 11              CBA
00190 01A1 26 1C           BNE    BTERR       NOT ENOUGH PARAMETERS
00191 01A3 FE A061 BTLOOP  LDX    BTINFO      POINTER FROM
00192 01A6 BC A063         CPX    BTINFO+2      STOP POINT
00193 01A9 26 01           BNE    BTARND
00194 01AB 5F              CLR B               IF NOT EQUAL B HAS 3
00195 01AC A6 00    BTARND LDA A  X           DATA
00196 01AE 08              INX                 INCREMENT FROM POINTER
00197 01AF FF A061         STX    BTINFO
00198 01B2 FE A065         LDX    BTINFO+4      WHERE POINTER
00199 01B5 A7 00           STA A  X           MOVE DATA
00200 01B7 08              INX
00201 01B8 FF A065         STX    BTINFO+4      INCREMENT WHERE REGISTER
00202 01BB 5D              TST B               FLAGS
00203 01BC 26 E5           BNE    BTLOOP
00204 01BE 39              RTS                 NORMAL EXIT
00205 01BF 86 04    BTERR  LDA A  #4
00206 01C1 7E 0260         JMP    SYSERR      ERROUT
```

```
00208                    ***PARSCN - SCAN CONSOLE FOR HEX PARAMETERS
00210                    *       INPUT   B   # OF PARAMETERS TO BE READ
00211                    *               X   WHERE PARAMETERS SHOULD BE STORED
00212                    *       OUTPUT  B   NUMBER PARAMETERS ENTERED BEFORE
00213                    *       REGISTER AFFECTED   B,CC
00214                    *       CLOSED SUBROUTINE?  YES, IF NO ERRORS.
00215                    *                           IF ERROR THEN GO TO DBUG
00216                    *       SUPPORT ROUTINES    RDCON,OUTCON,READCH,OUTCH,
00217                    *                           INBYTE,UNHEX,OHEX,OBYTE
00219 01C4 36            PARSCN PSH A
00220 01C5 F7 A055              STA B   COUNT1      #PARAMETERS TO BE READ
00221 01C8 7F A056              CLR     COUNT2      #PARAMETERS READ
00222 01CB 86 FF                LDA A   #-1         FLAG FOR CONSOLE
00223 01CD FF A05B PARSNO       STX     XTEMP1
00224 01D0 7D A055              TST     COUNT1
00225 01D3 27 4E                BEQ     THRUER      TOO MANY PARAMETERS IN TO COL
00226 01D5 CE A057              LDX     #PARBUF
00227 01D8 6F 00                CLR     0,X         CLEAR BUFFER AREA
00228 01DA 6F 01                CLR     1,X
00229 01DC 6F 02                CLR     2,X
00230 01DE 6F 03                CLR     3,X
00231 01E0 CE 0204 PARLP        LDX     #THANDL     ADDRESS OF NON-HEX HANDLER
00232 01E3 BD 027F              JSR     INHEX       INPUT 1 CHAR INTO LSH
00233 01E6 37                   PSH B               STORE HEX CHAR
00234 01E7 58                   ASL B               SHIFT TO MSH
00235 01E8 58                   ASL B
00236 01E9 58                   ASL B
00237 01EA 58                   ASL B
00238 01EB BD 0379              JSR     OHEX        ECHO
00239 01EE 33                   PUL B
00240 01EF 36                   PSH A               STORE CONSOLE FLAG
00241 01F0 CE A057              LDX     #PARBUF
00242 01F3 A6 02                LDA A   2,X         RIPPLE PAPRAMETER BUFFER
00243 01F5 A7 03                STA A   3,X
00244 01F7 A6 01                LDA A   1,X
00245 01F9 A7 02                STA A   2,X
00246 01FB A6 00                LDA A   0,X
00247 01FD A7 01                STA A   1,X
00248 01FF E7 00                STA B   0,X         LATEST HEX CHAR
00249 0201 32                   PUL A
00250 0202 20 DC                BRA     PARLP       NEXT CHARACTER
00252                    *       NON HEX HANDLER PARAMETER SEPARATION OR TERMIN
00254 0204 31            THANDL INS                  INCREMENT OVER INHEX RET ADR
00255 0205 31                   INS
00256 0206 36                   PSH A               FLAG FOR CONSOLE IN OTHER SEC
00257 0207 FE A05B              LDX     XTEMP1      RESTORE X
00258 020A BD 0344              JSR     OUTCON      ECHO NO HEX CHAR
00259 020D C1 2C                CMP B   #',         SEPARATION
```

```
00260 020F 26 08          BNE     THANOU    NO
00261 0211 8D 1D          BSR     RAPAR     RETURN A PARAMETER
00262 0213 7A A055        DEC     COUNT1    ONE LESS TO BE READ
00263 0216 32             PUL A
00264 0217 20 B4          BRA     PARSNO

00266 0219 C1 0D    THANOU CMP B  #$0D      CARRIAGE RETURN
00267 021B 27 04          BEQ     THAYSO    WAS C/R

00269                     *       ERROR IN ENTRY
00270 021D 86 02          LDA A   #2        ERROR TYPE 2
00271 021F 20 3F          BRA     SYSERR    OUTPUT EMSG AND RETURN SYSTEM

00273                     * PARAMETER SCAN TERMINATED BY C/R

00275 0221 8D 0D    THAYSO BSR    RAPAR     RETURN LAST PARAMETER

00277                     *       ERROR TOO MANY PARAMETERS RETURN FIRST RCV'D

00279 0223 F6 A056  THROUGH LDA B  COUNT2
00280 0226 32             PUL A             OSSOLE FLAG
00281 0227 32             PUL A             USER'S A REG
00282 0228 09       RPOX  DEX               REPOSITION X TO START
00283 0229 09             DEX
00284 022A 7A A056        DEC     COUNT2
00285 022D 26 F9          BNE     RPOX
00286 022F 39             RTS               END SCAN

00288                     *       RETURN A PARAMETER

00290 0230 7C A056  RAPAR INC     COUNT2    ONE MORE READ
00291 0233 B6 A05A        LDA A   PARBUF+3    MSH OF MSB OF PARAMETER
00292 0236 48             ASL A
00293 0237 48             ASL A
00294 0238 48             ASL A
00295 0239 48             ASL A
00296 023A BB A059        ADD A   PARBUF+2    LSH OF MSB OF PARAMETER
00297 023D A7 00          STA A   X         MSB
00298 023F 08             INX
00299 0240 B6 A058        LDA A   PARBUF+1    MSH OF LSB OF PARAMETER
00300 0243 48             ASL A
00301 0244 48             ASL A
00302 0245 48             ASL A
00303 0246 48             ASL A
00304 0247 BB A057        ADD A   PARBUF    LSH OF LSB OF PARAMETER
00305 024A A7 00          STA A   X         LSB
00306 024C 08             INX
00307 024D 39             RTS
```

```
00309                   ***DISINT - DISABLE INTERRUPT FROM
00310                   *       TTY RECEIVE PORTION
00311                   *        OF INTERFACES ONLY)

00313                   *        INPUT    N/A
00314                   *        OUTPUT   N/A
00315                   *        REGISTERS AFFECTED   CC ONLY
00316                   *        CLOSED SUBROUTINE?   YES
00317                   *        SUPPORT ROUTINES   N/A

00319 024E 36           DISINT  PSH A
00320 024F 86 11                LDA A  #TTYID
00321 0251 B7 8010              STA A  TTYCON
00322 0254 32                   PUL A
00323 0255 39                   RTS



00325                   ***ENINT - ENABLE INTERRUPT FROM CONSOLE  FEATURE.
00326                   *       TTY IS CONSOLE KEYBOARD
00327                   *       INPUT  X CONTAINS ADDRESS OF
00328                   *           WHERE TO GO AFTER INTERRUPT
00329                   *           FEATURE ENABLED.
00330                   *       OUTPUT  N/A
00331                   *       REGISTERS AFFECTED  CC ONLY
00332                   *       CLOSED SUBROUTINE?   DEPENDS ON [X]
00333                   *       SUPPORT ROUTINES   DEPENDENT ON [X]

00335 0256 36           ENINT   PSH A             STORE USER A-REG
00336 0257 86 91        CTTYOU  LDA A  #TTYIE     SET UP INT FOR TTY
00337 0259 B7 8010              STA A  TTYCON
00338 025C 32           ENIRND  PUL A
00339 025D 0E                   CLI
00340 025E 6E 00                JMP    X
```

```
00342                   ***SYSERR - SYSTEM ERROR HANDLER
00343                   *       OUTPUTS ERROR TYPE TO CONSOLE
00345                   *       INPUT A   ERROR TYPE NUMBER
00346                   *       OUTPUT    N/A
00347                   *       REGISTERS AFFECTED   ALL
00348                   *       CLOSED SUBROUTINE?   NO,RESTARTS SYSTEM
00349                   *       SUPPORT ROUTINES   CPSTNG,OBYTE,GBUG
00351                   ***SYSTEM ERRORS***
00352                   *       1   INVALID COMMAND
00353                   *       2   INVALID ARGUMENT
00354                   *       3   INVALID CHARACTER OR WRONG CHECKSUM ON LO
00355                   *       4   TOO FEW PARAMETERS
00356                   *       5   ATTEMPT TO MEMORY EXHANGE ON NON-RAM
00358 0260 36           SYSERR  PSH A           SYSTEM ERROR NUMBER
00359 0261 CE 0591              LDX   #SYSMSG
00360 0264 86 FF                LDA A #-1       CONSOLE
00361 0266 BD 03A2              JSR   CPSTNG    PRINT FIRST PART OF STRING
00362 0269 33                   PUL B           SYSTEM ERROR NUMBER
00363 026A BD 0391              JSR   OBYTE     OUTPUT NUMBER
00364 026D 7E 010F              JMP   SYSABT    SYSTEM ABORT
00366                   ***INBYTE - INPUT A BYTE FROM I/O DEVICE
00368                   *       INPUT   A < 0   INPUT FROM CONSOLE
00369                   *               A = 0   INPUT FROM READER
00370                   *               A > 0   INPUT FROM READER
00371                   *               A   ADDRESS OF ERROR HANDLER
00372                   *       OUTPUT  B   BYTE INPUTTED
00373                   *           (SEE INHEX FOR OUTPUT IF ERROR)
00374                   *       REGISTERS AFFECTED   B,CC
00375                   *       CLOSED SUBROUTINE?    YES , IF NO ERROR
00376                   *       SUPPORT ROUTINES   INHEX,ERROR,BUFCON,READCH
00378 0270 36           INBYTE  PSH A           STORE A
00379 0271 8D 0C                BSR   INHEX     INPUT MSH
00380 0273 58                   ASL B           MOVE WHAT IS MSH TO MSH
00381 0274 58                   ASL B
00382 0275 58                   ASL B
00383 0276 58                   ASL B
00384 0277 37                   PSH B           STORE MSH,0
00385 0278 8D 05                BSR   INHEX     INPUT LSH
00386 027A 32                   PUL A           RETRIEVE MSH,0 NOW IN A
00387 027B 1B                   ABA             COMBINE
```

```
00388 027C 16              TAB                 MOVE BACK TO B
00389 027D 32              PUL A               RESTORE A
00390 027E 39              RTS
00392                ***INHEX - INPUT ONE HEX CHARACTER.

00394                *      INPUT   A < 0   INPUT FROM CONSOLE
00395                *              A = 0   INPUT FROM READER
00396                *              A > 0   INPUT FROM READER
00397                *              X   ADDRESS OF ERROR HANDLER.
00398                *      OUTPUT   B   IF VALID INPUT:
00399                *                    HEX CHAR IN LS HALF (0 IN MSH)
00400                *                   IF INVALID INPUT:
00401                *                    ASCII CHARACTER INPUTTED.
00402                *      REGISTERS AFFECTED B,CC (X IF ERROR OCCURS AND
00403                *                    ROUTINE WAS CALLED BY PARSCN
00404                *      CLOSED SUBROUTINE   YES IF NO ERRORS
00405                *      SUPPORT ROUTINES   RDBR,RDCON,READCH

00407 027F 36        INHEX  PSH A               STORE LOGICAL DEVICE INFO
00408 0280 4D              TST A               SET FLAGS
00409 0281 2D 04           BLT    RDCON1       BRANCH IF FROM CONSOLE
00410 0283 8D 1B           BSR    RDBR         READER READ REQUEST
00411 0285 20 02           BRA    INHEX0       REJOIN LOGIC
00412 0287 8D 20     RDCON1 BSR    RDCON
00413 0289 32        INHEX0 PUL A               RESTORE
00414 028A 37              PSH B
00415 028B C0 30           SUB B  #$30         REMOVE ASCII BIAS
00416 028D 2D 0E           BLT    ILCHR0       NOT HEX
00417 028F C1 09           CMP B  #9
00418 0291 2F 08           BLE    INCHOK       0-9
00419 0293 C0 07           SUB B  #$7          MORE BIAS
00420 0295 2D 06           BLT    ILCHR0       NOT HEX
00421 0297 C1 0F           CMP B  #$F
00422 0299 2E 02           BGT    ILCHR0       NOT HEX
00423 029B 31        INCHOK INS                 A-F AND RENTRY OF 0-9
00424                *                   MOVESTACK PAST ASCII CHAR (NOT NEEDE
00425                *                   TO GET RET ADDRESS ON TOP.S
00426 029C 39              RTS
00427 029D 33        ILCHR0 PUL B               ILLEGAL CHAR IN B
00428 029E 6E 00           JMP    X            JUMP USER SPECIFIED ERROR ROU
```

```
00430                   ***RRDR - READ FROM READER
00432                   *       INPUT   N/A
00433                   *       OUTPUT  A   CORRECT CODE FOR
00434                   *           READER AS PER [DEVICE] FOR
00435                   *           SUBROUTINE READCH.
00436                   *       REGISTERS AFFECTED   A,CC
00437                   *       CLOSED SUBROUTINE   NO, GOES DIRECTLY TO READ
00438                   *       SUPPORT ROUTINES   READCH

00440 02A0 4F       RRDR    CLR A
00441 02A1 7D A050          TST     DEVICE
00442 02A4 27 01            BEQ     RRDRYS      BRANCH IF TTY
00443 02A6 4C               INC A               IF PTR
00444 02A7 20 01    RRDRYS  BRA     READCH


00446                   ***RDCON - READ ONE ASCII CHAR FROM CONSOLE.
00448                   *       INPUT   N/A
00449                   *       OUTPUT   A = 0  TTY IS CONSOLE KEYBOARD
00450                   *       REGISTERS AFFECTED   A,CC
00451                   *       CLOSED SUBROUTINE?   NO, GOES DIRECT TO READC
00452                   *       SUPPORT ROUTINES   READCH

00454 02A9 4F       RDCON   CLR A
```

```
00456                    ***READCH - INPUT A CHARACTER FROM THE
00457                    *       PROPER DEVICE.
00459                    *       INPUT    A  <0 CAUSES READ FROM TTY/TVT AS DIS
00460                    *                A  =0 CAUSES READ FROM TTY
00461                    *                A  >0 CAUSES READ FROM PTR
00462                    *       OUTPUT   A  ASCII CHARACTER READ
00463                    *           FROM UNIT W/O PARITY BIT
00464                    *       REGISTERS AFFECTED   B,CC
00465                    *      CLOSED SUBROUTINE?    YES
00466                    *       SUPPORT ROUTINES     N/A

00468 02AA 4D       READCH TST A
00469 02AB 2B 02            BMI     RDTTY1    READ TVT/TTY KEYBOARD
00470 02AD 26 0D            BNE     RDHSTR    READ PTR

00472 02AF F6 8010 RDTTY1 LDA B   TTYCON    READ TTY READY?
00473 02B2 54             LSR B
00474 02B3 24 FA            BCC     RDTTY1    BRANCH IF NOT
00475 02B5 F6 8011          LDA B   TTYIOB    OTHERWISE READ
00476 02B8 20 1F            BRA     RCHOUT    STRIP PARITY AND RTS

00478 02BA 20 1D            BRA     RCHOUT    STRIP PARITY AND RTS

00480 02BC C6 60    RDHSTR LDA B   #START    START READER
00481 02BE F7 8022          STA B   PIACON
00482 02C1 F6 8022 RCH000 LDA B   PIACON    WAIT FOR FALL
00483 02C4 56             ROR B
00484 02C5 25 FA            BCS     RCH000
00485 02C7 F6 8022          LDA B   PIACON    CHECK TRUE LOW
00486 02CA 56             ROR B
00487 02CB 25 F4            BCS     RCH000    NOT TRUE LOW
00488 02CD F6 8022 RCH001 LDA B   PIACON
00489 02D0 56             ROR B
00490 02D1 24 FA            BCC     RCH001    WAIT FOR RISING EDGE
00491 02D3 F6 8020          LDA B   PIARD     READ
00492 02D6 7F 8022          CLR     PIACON    STOP READER

00494 02D9 C4 7F    RCHOUT AND B   #$7F      STRIP PARITY
00495 02DB 39               RTS
```

```
00497                    *****LOAD PROGRAM
00498 02DC 7D A050 LOADER TST     DEVICE
00499 02DF 26 09          BNE     LOAD        TTY READER,NO IRQ
00500 02E1 CE 02EA        LDX     #LOAD
00501 02E4 7E 0256        JMP     ENTINT      TTY NOT READER ENABLE IRQ
00502 02E7 7E 0256        JMP     ENTINT
00503 02EA 8D B4    LOAD  BSR     RDBR
00504 02EC C1 53          CMP B   #'S
00505 02EE 26 FA          BNE     LOAD
00506 02F0 8D B8          BSR     READCH      A ALREADY SET UP
00507 02F2 C1 39          CMP B   #'9
00508 02F4 27 1D          BEQ     LSTOP
00509 02F6 C1 31          CMP B   #'1
00510 02F8 26 F0          BNE     LOAD
00511 02FA 7F A05F        CLR     CKSUM
00512 02FD 4F             CLR A                 READER
00513 02FE 8D 2F          BSR     BYCK        # OF BYTES UPDATE CKSUM
00514 0300 C0 02          SUB B   #2
00515 0302 F7 A060        STA B   BYCNT       BYTE COUNT
00516 0305 8D 1A          BSR     FILLX       COMPUTE START ADDRESS
00517 0307 8D 26    LOAD2 BSR     BYCK        INPUT DATA
00518 0309 7A A060        DEC     BYCNT
00519 030C 27 06          BEQ     LOAD3       DONE
00520 030E E7 00          STA B   X           STORE DATA
00521 0310 08             INX
00522 0311 20 F4          BRA     LOAD2
00523 0313 39       LSTOP RTS
00524 0314 7C A05F LOAD3  INC     CKSUM
00525 0317 27 D1          BEQ     LOAD
00526 0319 7F 8022 OUT    CLR     PIACOM      READER OFF
00527 031C 86 03          LDA A   #3          ERROR 3
00528 031E 7E 0260        JMP     SYSERR
00529 0321 8D 0C    FILLX BSR     BYCK        FILL X WITH ADDR
00530 0323 F7 A05D        STA B   XMSB
00531 0326 8D 07          BSR     BYCK
00532 0328 F7 A05E        STA B   XLSB
00533 032B FE A05D        LDX     XMSB
00534 032E 39             RTS
00535 032F FF A05B BYCK   STX     XTEMP1
00536 0332 CE 0319        LDX     #OUT        ERROR HANDLING
00537 0335 BD 0270        JSR     INBYTE
00538 0338 37             PSH B
00539 0339 FB A05F        ADD B   CKSUM
00540 033C F7 A05F        STA B   CKSUM
00541 033F 33             PUL B
00542 0340 FE A05B        LDX     XTEMP1      RESTORE X
00543 0343 39             RTS
```

```
00545                    ***OUTCON - OUTPUT 1 ASCII CHAR TO CONSOLE.
00547                    *       INPUT   B   CHARACTER TO BE OUTPUT
00548                    *       OUTPUT  A   0 IF TTY IS CONSOLE.
00549                    *                  -1 IF TVT IS CONSOLE.
00550                    *       REGISTERS AFFECTED   A,CC
00551                    *       CLOSED SUBROUTINE?   NO  GOES DIRECT TO OUTCH
00552                    *       SUPPORT ROUTINES     OUTCH

00554 0344 4F       OUTCON CLR A
00555 0345 7D A050         TST     DEVICE    DETERMINE CONSOLE
00556 0348 27 01           BEQ     OUTCH     BRANCH IF TTY
00557 034A 4A              DEC A             TVT


00559                    ***OUTCH - OUTPUT ONE ASCII CHARACTER TO
00560                    *       TTY OR TVT

00562                    *       INPUT   A  <0 OUTPUT TO TVT
00563                    *                  =0 OUTPUT TO TTY
00564                    *                  >0 OUTPUT TO TVT
00565                    *       OUTPUT  N/A
00566                    *       REGISTERS AFFECTED   CC ONLY
00567                    *       CLOSED SUBROUTINE?   YES
00568                    *       SUPPORT ROUTINES     N/A

00570 034B 4D       OUTCH  TST A
00571 034C 27 0C           BEQ     OUTTTY    BRANCH FOR TTY

00573 034E 7D 800B OUTTV1 TST     PTVTC2    TVT READY?
00574 0351 2A FB           BPL     OUTTV1    BRANCH IF NOT
00575 0353 7D 800A         TST     PTVTWR    CLEAR FLAG
00576 0356 F7 800A         STA B   PTVTWR    OUTPUT
00577 0359 39              RTS

00579 035A 36       OUTTTY PSH A
00580 035B B6 8010 TTYL00 LDA A   TTYCON    GET B2 INTO CARRY
00581 035E 46              ROR A
00582 035F 46              ROR A
00583 0360 24 F9           BCC     TTYL00    BRANCH TTY NOT READY
00584 0362 F7 8011         STA B   TTYIOL    OUTPUT
00585 0365 32              PUL A
00586 0366 39              RTS
```

```
00588                    ***OPUNCH - OUTPUT TO PUNCH
00589                    *       INPUT    B    ASCII CHAR TO BE OUTPUT.
00590                    *      OUTPUT    A     0 TO SIGNAL PUNCH AS TTY
00591                    *       REGISTERS AFFECTED    A,CC
00592                    *       CLOSED SUBROUTINE?    NO, GOES DIRECT TO OUTCH
00593                    *       SUPPORT ROUTINES    OUTCH

00595 0367 4F            OPUNCH CLR A              TTY
00596 0368 20 E1                BRA     OUTCH


00598                    ***NULLS - OUTPUT 60 NULLS TO PUNCH

00600                    *       INPUT    N/A
00601                    *       OUTPUT    N/A
00602                    *       REGISTERS AFFECTED    CC ONLY
00603                    *       CLOSED SUBROUTINE?    YES
00604                    *       SUPPORT ROUTINES    OPUNCH

00606 036A 36            NULLS   PSH A
00607 036B 37                    PSH B
00608 036C 5F                    CLR B
00609 036D 86 3C                 LDA A   #60
00610 036F 36            NULLLS  PSH A             STORE A
00611 0370 8D F5                 BSR     OPUNCH    OUTPUT A NULL
00612 0372 32                    PUL A
00613 0373 4A                    DEC A
00614 0374 26 F9                 BNE     NULLLS    DO 60 TIMES
00615 0376 33                    PUL B
00616 0377 32                    PUL A
00617 0378 39                    RTS
```

```
00619                    ***OHEX - OUTPUT MOST SIGNIFICANT HEX CHARACTER
00620                    *       OF B REGISTER
00622                    *       INPUT   A< 0  OUTPUT OT CONSOLE.
00623                    *               A= 0  OUTPUT OT LIST.
00624                    *               A> 0  OUTPUT TO  PUNCH.
00625                    *               B   BYTE TO WITH MS HSLF TO BE OUTPUT.
00626                    *       OUTPUT    1 ASCII OF MS HALF WHICH WAS OUTPUTTE
00627                    *       REGISTERS AFFECTED    A,B,CC
00628                    *       CLOSED SUBROUTINE?    NO,OPEN ENDED TO RETURN F
00629                    *       SUPPORT ROUTINES      OUTCON,OPUNCH,OUTLST,OUTCH.

00631 0379 C4 F0    OHEX    AND B   #$F0        GET MS HALF
00632 037B 54               LSR B               MOVE TO LS HALF
00633 037C 54               LSR B
00634 037D 54               LSR B
00635 037E 54               LSR B
00636 037F CB 30            ADD B   #$30        ADD BIAS
00637 0381 C1 39            CMP B   #$39
00638 0383 2F 02            BLE     OHEX00      BRANCH 0-9
00639 0385 CB 07            ADD B   #$7         A-F ADD ADDITIONAL BIAS
00640 0387 4D       OHEX00  TST A               DETERMINE LOGICAL OUTPUT
00641 0388 2B BA            BMI     OUTCON      CONSOLE
00642 038A 27 02            BEQ     OUTLST      LIST
00643 038C 20 09            BRA     OPUNCH      PUNCH


00645                    ***OUTLST - OUTPUT 1 ASCII CHAR TO LIST DEVICE.
00647                    *       INPUT   B   ASCII CHAR TO OUTPUT
00648                    *       OUTPUT  A 0 (TO DESIGNATE TTY)
00649                    *       REGISTERS AFFECTED    A,CC
00650                    *       CLOSED SUBROUTINE?    NO, GOES DIRECT TO OUTCH.
00651                    *       SUPPORT ROUTINES      OUTCH

00653 038E 4F       OUTLST  CLR A
00654 038F 20 BA            BRA     OUTCH
```

```
00656                   ***OBYTE - OUTPUT BYTE (2 HEX CHAR)
00658                   *       INPUT   A  <0  OUTPUT TO CONSOLE.
00659                   *               A  =0  OUTPUT TO LIST.
00660                   *               A  >0  OUTPUT TO PUNCH.
00661                   *               B   BYTE TO BE OUTPUT
00662                   *       OUTPUT    N/A
00663                   *       REGISTERS AFFECTED CC
00664                   *       CLOSED SUBROUTINE?   YES
00665                   *       SUPPORT ROUTINES   OHEX,OUTCON,OPUNCH,OUTLST,O

00667 0391 36       OBYTE  PSH A            STORE A,B
00668 0392 37              PSH B
00669 0393 36              PSH A
00670 0394 37              PSH B
00671 0395 8D E2           BSR    OHEX     MS HALF
00672 0397 33              PUL B           RESTORE A,B
00673 0398 32              PUL A
00674 0399 58              ASL B
00675 039A 58              ASL B
00676 039B 58              ASL B
00677 039C 58              ASL B           LS NOW MS
00678 039D 8D DA           BSR    OHEX     LS HALF
00679 039F 33              PUL B           RESTORE REGISTERS
00680 03A0 32              PUL A
00681 03A1 39              RTS
```

```
00683                    ***CPSTNG - PRINT CHARACTER STRING
00684                    *       TERMINATED BY 'EOT' (04H)
00686                    *       INPUT    A < 0   OUTPUT TO CONSOLE
00687                    *                A = 0   OUTPUT TO LIST
00688                    *                A > 0   OUTPUT TO PUNCH
00689                    *                X   START OF CHAR STRING
00690                    *       OUTPUT   X   END OF STRING + 1
00691                    *       REGISTERS AFFECTED   X,CC
00692                    *       CLOSED SUBROUTINE?   YES
00693                    *       SUPPORT ROUTINES   OUTCON,OUTLST,OPUNCH,OUTCH
00695 03A2 37            CPSTNG PSH B
00696 03A3 36            CPSTLP PSH A
00697 03A4 E6 00                LDA B  X          CHAR
00698 03A6 08                   INX
00699 03A7 C1 04                CMP B  #4         EOT?
00700 03A9 27 1A                BEQ    CPYES      YES,OUT
00701 03AB 4D                   TST A             NO,COMPUTE DEVICE
00702 03AC 2B 0A                BMI    CPCON      PRINT CHAR ON CONSOLE
00703 03AE 27 04                BEQ    CPLST      PRINT CHAR ON LIST
00704 03B0 8D 85                BSR    OPUNCH     PUNCH CHAR
00705 03B2 20 06                BRA    CPSTOO
00706 03B4 8D 08          CPLST BSR    OUTLST     LIST CHAR
00707 03B6 20 02                BRA    CPSTOO
00708 03B8 8D 8A          CPCON BSR    OUTCON     WRITE ON CONSOLE
00710                    *     ADD 50 MS DELAY IF OUTPUT WAS IVT ERASE COMMAND
00712 03BA C1 15          CPSTOO CMP B  #$15
00713 03BC 27 0A                BEQ    CDELAY
00714 03BE C1 16                CMP B  #$16
00715 03C0 27 06                BEQ    CDELAY
00716 03C2 32                   PUL A             RESTORE INPUTTED A
00717 03C3 20 DE                BRA    CPSTLP
00718 03C5 32             CPYES PUL A             RESTORE INPUTTED A
00719 03C6 33                   PUL B
00720 03C7 39                   RTS
00721 03C8 86 19          CDELAY LDA A  #25
00722 03CA 36             CPAGN0 PSH A
00723 03CB 4F                   CLR A
00724 03CC 4A             CPAGN1 DEC A
00725 03CD 26 FD                BNE    CPAGN1
00726 03CF 32                   PUL A
00727 03D0 4A                   DEC A
00728 03D1 26 F7                BNE    CPAGN0
00729 03D3 32                   PUL A
00730 03D4 20 CD                BRA    CPSTLP
00731 03D6 CE A06D PRINT  LDX    #MONOFF
00732 03D9 C6 02                LDA B  #2
00733 03DB BD 0164              JSR    PARSCN     ACQUIRE PRINT START,STOP
00734 03DE C1 02                CMP B  #2
```

```
00735 03E0 27 05          BEQ     POK0
00736 03E2 86 04          LDA A   #4
00737 03E4 7E 0260        JMP     SYSERR    ERROR,NOT ENOUGH PARAMETER
00738 03E7 CE 03ED POK0   LDX     #POK1     ENABLE INT
00739 03EA 7E 0256        JMP     EN.INT
00740 03ED CE 0598 POK1   LDX     #CRLF
00741 03F0 86 01          LDA A   #1        PUNCH
00742 03F2 BD 03A2        JSR     CPSTNG
00743 03F5 8D 03A2        JSR     CPSTNG    HEADER
00744 03F8 FE A06D PRNT   LDX     DBUFF     FRAME START ADDRESS
00745 03FB B6 A070        LDA A   DBUFF+3    LSB'S
00746 03FE B0 A06E        SUB A   DBUFF+1
00747 0401 F6 A06E        LDA B   DBUFF+1
00748 0404 F6 A06F        LDA B   DBUFF+2    MSB'S
00749 0407 F2 A06D        SBC B   DBUFF
00750 040A 26 04          BNE     PRNT00    >16 BYTES LEFT
00751 040C 81 10          CMP A   #16
00752 040E 25 02          BCS     PRNT01    <16 BYTES LEFT
00753 0410 86 0F PRNT00   LDA A   #15       MAX FRAME SIZE
00754 0412 8B 04 PRNT01   ADD A   #4        ADDRESS,BYTE COUNT,CKSUM
00755 0414 16             TAB               B HAS FRAME COUNT
00756 0415 7F A05F        CLR     CKSUM
00757 0418 8D 043F        JSR     O1BYT     OUTPUT FRAME COUNT
00758 041B F6 A06D        LDA B   DBUFF
00759 041E 8D 1F          BSR     O1BYT     MXB ADDRESS
00760 0420 F6 A06E        LDA B   DBUFF+1
00761 0423 8D 1A          BSR     O1BYT     LSB ADDRESS
00762 0425 80 03          SUB A   #3        THREE LESS IN FRAME COUNT
00763 0427 E6 00 PRNT02   LDA B   X
00764 0429 8D 14          BSR     O1BYT     OUTPUT DATA BYTE
00765 042B 08             INX
00766 042C 4A             DEC A
00767 042D 26 F8          BNE     PRNT02
00768 042F F6 A05F        LDA B   CKSUM
00769 0432 53             COM B             OUTPUT CKSUM
00770 0433 8D 0A          BSR     O1BYT
00771 0435 FF A06D        STX     DBUFF     NEXTFRAME'S START
00772 0438 09             DEX
00773 0439 BC A06F        CPX     DBUFF+2    DONE?
00774 043C 26 AF          BNE     POK1      NO,NEXT FRAME
00775 043E 39             RTS               DONE
00776 043F 8D 0391 O1BYT  JSR     OUTBYTE   OUTPUT BYTE
00777 0442 FB A05F        ADD B   CKSUM     UPDATE CKSUM
00778 0445 F7 A05F        STA B   CKSUM
00779 0448 39             RTS
00780 0449 CE A06D DISPLY LDX     #DBUFF
00781 044C C6 02          LDA B   #2
00782 044E 8D 01C4        JSR     PARSCH
00783 0451 C1 02          CMP B   #2
00784 0453 27 0A          BEQ     DEP
00785 0455 A6 00          LDA A   X         MAKE START AND STOP SAME
00786 0457 A7 02          STA A   2,X
```

```
00787 0459 A6 01          LDA A  1,X
00788 045B A7 03          STA A  3,X
00789 045D 86 FF          LDA A  #-1
00790 045F 20 2B    DLP   BRA    NWLINE
00791 0461 FE A06D DISP00 LDX    DBUFF     X HAS START ADDRESS
00792 0464 86 F0          LDA A  #-16      COUNT ALWAYS NEG FOR CONSOLE
00793 0466 F6 A06D        LDA B  DBUFF
00794 0469 BD 0391        JSR    OUTBYTE   ADDRESS MSB
00795 046C F6 A06E        LDA B  DBUFF+1
00796 046F BD 0391        JSR    OUTBYTE   ADDRESS LSB
00797 0472 C6 20    DISP02 LDA B  #$20      SPACE
00798 0474 36             PSH A
00799 0475 BD 0344        JSR    OUTCON
00800 0478 32             PUL A
00801 0479 E6 00          LDA B  0,X        BYTE
00802 047B BD 0391        JSR    OUTBYTE
00803 047E BC A06F        CPX    DBUFF+2
00804 0481 26 01          BNE    DSPNO
00805 0483 39             RTS               DONE
00806 0484 08       DSPNO INX
00807 0485 4C             INC A             COUNT
00808 0486 26 EA          BNE    DISP02     NEXT BYTE
00809 0488 FF A06D        STX    DBUFF      UPDATE DBUFF
00810 048B 4A             DEC A             CONSOLE
00811 048C CE 0598 NWLINE LDX    #CRLF
00812 048F BD 03A2        JSR    OUTSTNG    NEW LINE
00813 0492 20 CD          BRA    DISP00
00814 0494 C6 01    GOTO  LDA B  #1
00815 0496 CE A06B        LDX    #GBUF
00816 0499 BD 01C4        JSR    PARSCN
00817 049C 5D             TST B
00818 049D 27 0D          BEQ    GOTOD
00819 049F FE A008        LDX    TSTACK
00820 04A2 B6 A06B        LDA A  GBUF
00821 04A5 A7 06          STA A  6,X
00822 04A7 B6 A06C        LDA A  GBUF+1
00823 04AA A7 07          STA A  7,X
00824 04AC CE 04B5 GOTOD  LDX    #GOTONW
00825 04AF DE A008        LDS    TSTACK
00826 04B2 7E 0256        JMP    ENINT
00827 04B5 3B       GOTONW RTI
```

```
00829                    *       ENTRY POINT FOR M COMMAND
00831                    *       INPUT    N/A
00832                    *       OUTPUT   N/A
00833                    *       REGISTER AFFECTED    ALL
00834                    *       CLOSED SUBROUTINE?   YES,NO ENTRY ERRORS
00835                    *       SUPPORT ROUTINES     PARSCN,OUTBYTE,INBYTE
00837 04B6 CE A069 MEXCHG LDX     #MBUF01    BUFFER
00838 04B9 C6 01          LDA B   #1         1 PARAMETER
00839 04BB BD 01C4        JSR     PARSCN     GET ADDRESS
00840 04BE CE 0598        LDX     #CRLF
00841 04C1 86 FF          LDA A   #-1
00842 04C3 BD 03A2        JSR     OUTSTNG    NEW LINE
00843 04C6 FE A069        LDX     MBUF01     X GETS ADDRESS
00844 04C9 8D 02    MEXLOP BSR    BOARIO     DO OPERATION
00845 04CB 20 FC          BRA     MEXLOP     REPEAT
```

```
00847                    ***BOARIO - BYTE ORIENTED ARGUMENT I/O
00848                    *       WRITE OR CONSOLE AND READ FROM CONSOLE.
00850                    *       INPUT    A  BYTE TO BE USED
00851                    *       OUTPUT   A  X+1 IF HEX # OR SPACE ENTERED
00852                    *       REGISTERS AFFECTED   ALL
00853                    *       CLOSE SUBROUTINE   DEPENDENT ON ENTRYS
00854                    *       SUPPORT ROUTINES   OUTCON,OBYTE,INHEX,SYSERR
00855 04CD C6 20    BOARIO LDA B  #$20      SPACE
00856 04CF BD 0344         JSR    OUTCON    OUTPUT SPACE
00857 04D2 4A              DEC A             CONSOLE FLAG
00858 04D3 C6 00           LDA B  #0
00859 04D5 BD 0391         JSR    OBYTE     OUTPUT BYTE
00860 04D8 C6 2D           LDA B  #'-
00861 04DA BD 0344         JSR    OUTCON
00862 04DD FF A071         STX    BOTMP1
00863 04E0 CE 050B         LDX    #BEVAL    NON HEX EVAL
00864 04E3 4A              DEC A             CONSOLE
00865 04E4 BD 027F         JSR    INHEX     MSH
00866 04E7 58              ASL B
00867 04E8 58              ASL B
00868 04E9 58              ASL B
00869 04EA 58              ASL B
00870 04EB 37              PSH B             STORE MSH
00871 04EC BD 0379         JSR    ONEX      ECHO
00872 04EF CE 0522         LDX    #BERR2
00873 04F2 BD 027F         JSR    INHEX
00874 04F5 37              PSH B             LSH H
00875 04F6 58              ASL B
00876 04F7 58              ASL B
00877 04F8 58              ASL B
00878 04F9 58              ASL B
00879 04FA BD 0379         JSR    ONEX      ECHO
00880 04FD 32              PUL A
00881 04FE 33              PUL B
00882 04FF 1B              ABA               BYTE IN A
00883 0500 FE A071         LDX    BOTMP1
00884 0503 A7 00           STA A  X         STORE IT
00885 0505 A1 00           CMP A  X         CHECK IT
00886 0507 26 1E           BNE    BERR5     ERR OUT IF NOT RAM
00887 0509 08              INX
00888 050A 39              RTS
00889 050B 31       BEVAL  INS               INHEX RET ADR
00890 050C 31              INS
00891 050D 36              PSH A
00892 050E BD 0344         JSR    OUTCON    ECHO
00893 0511 32              PUL A
00894 0512 C1 20           CMP B  #$20      SPACE
00895 0514 26 05           BNE    BEVAL2
00896 0516 FE A071         LDX    BOTMP1    WAS SPACE
00897 0519 08              INX               MAKE NO CHANGE
00898 051A 39              RTS               INC X AND RET
```

```
00899 0518 C1 0D    BEVAL2  CMP B   #$0D        C/R
00900 051A 26 03            BNE     BERR2
00901 051F 31               INS                 OVER MEX OR REG CALLING RET A
00902 0520 31               INS
00903 0521 39               RTS                 TO MAJOR ROUTINE
00904 0522 86 02    BERR2   LDA A   #2
00905 0524 7E 0260  BERROT  JMP     SYSERR
00906 0527 86 05    BERR5   LDA A   #5
00907 0529 20 F9            BRA     BERROT
00908 052B BD 02A9  RECANG  JSR     RCCON       READ CHAR
00909 052E BD 0344          JSR     OUTCON      ECHO
00910 0531 FE A008          LDX     #STACK      USER SP TO X
00911 0534 08               INX                 POINT TO CC
00912 0535 C1 43            CMP B   #'C
00913 0537 27 24            BEQ     REG8BT
00914 0539 08               INX                 POINT OT B
00915 053A C1 42            CMP B   #'B
00916 053C 27 1F            BEQ     REG8BT
00917 053E 08               INX                 POINT TO A
00918 053F C1 41            CMP B   #'A
00919 0541 27 1A            BEQ     REG8BT
00920 0543 08               INX                 POINT TO X
00921 0544 C1 58            CMP B   #'X
00922 0546 27 12            BEQ     REG16B
00923 0548 08               INX
00924 0549 08               INX                 POINT TO PC
00925 054A C1 50            CMP B   #'P
00926 054C 27 0C            BEQ     REG16B
00927 054E CE A008          LDX     #STACK      STORAGE FOR USER SP
00928 0551 C1 53            CMP B   #'S
00929 0553 27 05            BEQ     REG16B
00930 0555 86 02            LDA A   #2          NOT REG
00931 0557 7E 0260          JMP     SYSERR
00932 055A BD 04CD  REG16B  JSR     BOARIO
00933 055D BD 04CD  REG8BT  JSR     BOARIO
00934 0560 39               RTS
```

```
00936                    ***MEOF - PUNCH EOF ON PAPER TAPE AND TRAILER

00938 0561 86 01    MEOF    LDA A   #1          PUNCH
00939 0563 CE 0598          LDX     #CRLF
00940 0566 BD 03A2          JSR     OPSTNG
00941 0569 CE 0578          LDX     #EOF
00942 056C BD 03A2          JSR     OPSTNG
00943 056F CE 0598          LDX     #CRLF
00944 0572 BD 03A2          JSR     OPSTNG
00945 0575 7E 036A          JMP     NULLS       NULLS AND RETURN
00946 0578 53       EOF     FCB     'S,'9,4
      0579 39
      057A 04
```

```
00948 057B 0A       GBMSG  FCB   $A,$D,$10,$16
      057C 0D
      057D 10
      057E 16
00949 057F 2A              FCC   /*GBUG 1.0*/
      0580 47
      0581 42
      0582 55
      0583 47
      0584 20
      0585 31
      0586 2E
      0587 30
      0588 2A
00950 0589 04              FCB   4
00951 058A 0D       SYSTNG FCB   $D,$A,,,,SYSSYM,4
      058B 0A
      058C 00
      058D 00
      058E 00
      058F 3E
      0590 04
00952 0591 0D       SYEMSG FCB   $D,$A,,,,'E,4
      0592 0A
      0593 00
      0594 00
      0595 00
      0596 45
      0597 04
00953 0598 0D       CRLF   FCB   $D,$A,,,,4
      0599 0A
      059A 00
      059B 00
      059C 00
      059D 04
00954 059E 53       HEADER FCB   'S,'1,4
      059F 31
      05A0 04
00955                      *****  CHANGE
00956 A050                 ORG   $A050
00957 A050 0001     DEVICE RMB   1
00958 A051 0001            RMB   1
00959 A052 0002     USRINT RMB   2
00960 A054 0001     BYTEMP RMB   1
00961 A055 0001     COUNT1 RMB   1
00962 A056 0001     COUNT2 RMB   1
00963 A057 0004     PARBUF RMB   4
00964 A05B 0002     XTEMP1 RMB   2
00965 A05D 0001     XMSB   RMB   1
00966 A05E 0001     XLSB   RMB   1
00967 A05F 0001     CKSUM  RMB   1
00968 A060 0001     BYCNT  RMB   1
```

```
00969 A061 0006     BTINFO RMB     6
00970 A067 0002     MBUFER RMB     2
00971 A069 0002     MBUF01 RMB     2
00972 A06B 0002     GBUF   RMB     2
00973 A06D 0004     DBUFF  RMB     4
00974 A071 0002     BOTMP1 RMB     2
00975      A008     TSTACK EQU     $A008
00976      A042     SYSSPO EQU     $A042
00977      02EA     TRACE  EQU     LOAD
00978      A030     ISTACK EQU     $A02D
00979      02EA     SWIHAN EQU     LOAD
00980      02EA     NMIHAN EQU     LOAD
00981 A048          ORG    $A048
00982 A048 00C4     FDB    GB00
00983               END
```


SYMBOL TABLE

```
POLL11 00B0  CONINT 00B8  INTOT1 00BB  GB00   00C4  PTVTRD 8008
PTVTWR 800A  PTVTC2 800B  TTYCON 8010  TTYIOL 8011  PIARD  8020
PIACON 8022  TTYIE  0091  TTYIB  0011  STARTR 0060  RWDRTR 0040
STPTR  0000  SYSSYM 005E  ABORT  0101  TRICK1 010C  SYSABT 010F
RENTRY 0117  TRICK2 0119  GBN01  0132  GBN02  0139  GBN03  0140
GBN04  0147  GBN05  0151  GBN06  0156  GBN07  015F  GBN08  016B
GBN09  016D  GBN010 0174  STAY   0183  GBN011 0185  GBN012 018C
ERR1   0195  BTRANS 0198  BTLOOP 01A3  BTARND 01AC  BTERR  01BF
PARSCN 01C4  PARSNO 01CD  PARLP 01E0  THANDL 0204  THAN00 0219
THAYS0 0221  THRUER 0223  RPOX   0228  RAPAR  0230  DISINT 024E
ENINT  0250  CTTY00 0257  ENIRND 025C  SYSERR 0260  INBYTE 0270
INHEX  027F  RDCONT 0287  INHEX0 028A  INCHCK 029B  ILCHR0 029D
RRDR   02A0  RRDRYS 02A7  RDCON  02A9  READCH 02AA  RDTTY1 02AF
RDHSTR 02BC  RCH000 02C1  RCH001 02CD  RCHOUT 02D9  LOADER 02DC
LOAD   02EA  LOAD2  0307  LSTOP  0313  LOAD3  0314  OUT    0319
FILLA  0321  BYCK   032F  OUTCON 0344  OUTCH  034B  OUTTVT 034E
OUTTTY 035A  TTYL00 035B  OPONCH 0367  NULLS  036A  NULLLS 036F
OHEX   0379  OHEX00 0387  OUTLST 038E  OBYTE  0391  CPSTNG 03A2
CPSTLP 03A5  CPLST  03B4  CPCON  03B5  CPST00 03BA  CPYES  03C5
DELAY  03C6  CPAGN0 03CA  CPAGN1 03CC  PRINT  03D6  PORO   03E7
POR1   03ED  PRNT   03F8  PRNT00 0410  PRNT01 0412  PRNT02 0427
DIBYT  043F  DISPLY 0449  DLP    045F  DISP00 0461  DISP02 0472
DSPN0  0484  NWLINE 048C  GOTO   0494  GOTO0  04AC  GOTONW 04B5
MEXCHG 04BB  MEXLOP 04C9  BOARD0 04CD  BEVAL  050B  BEVAL2 051B
BERR2  052E  BERROT 0524  BERRS  0527  REGXNG 052B  REG100 055A
REGBIT 055D  MEOF   0561  EOF    057B  GBMSG  057B  SYSTNG 058A
SYEMSG 0591  CRLF   0598  HEADER 059E  DEVICE A050  USEINT A052
BYTEMP A054  COUNT1 A055  COUNT2 A056  PARBUF A057  XTEMP1 A05B
XMSB   A05D  XLSB   A05E  CKSUM  A05F  BYCNT  A060  BTINFO A061
MBUFER A067  MBUF01 A069  GBUF  A06B  DBUFF  A06D  BOTMP1 A071
TSTACK A008  SYSSPO A042  TRACE  02EA  ISTACK A030  SWIHAN 02EA
NMIHAN 02EA
```

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (19) 13594.1-R-EL (18) ARO	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* (6) A MICROPROCESSOR SUBSYSTEM FOR AUTOMATIC TESTING.	5. TYPE OF REPORT & PERIOD COVERED (9) Final Report. 15 Oct 75 - 14 Jun 76.
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) Donald K. Fronek, David G. Green	8. CONTRACT OR GRANT NUMBER(s) (15) DAAG29-76-G-0062
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of Alabama, Huntsville, Alabama 35807	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS (12) 67p.
11. CONTROLLING OFFICE NAME AND ADDRESS U. S. Army Research Office, Post Office Box 12211, Research Triangle Park, NC 27709	12. REPORT DATE (11) August 1976
	13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*	15. SECURITY CLASS. *(of this report)* Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Computers
Computer programs
Interfaces
Microcomputers
Central processing unit
Logic circuits
Arithmetic functions

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

(see other side)

409893

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

The research effort has progressed in four parts:

1. Study existing and projected measurement requirements of the LCSS relative to a resident microprocessor.
2. Design and fabricate a microprocessor system capable of controlling the administrative and logic functions of the LCSS.
3. Develop corresponding software subroutines to support item 2 above.
4. Interface the microprocessor and perform functional testing.

This research can be adequately described in two categories - hardware and software. The hardware portion of this Report relates that effort of providing diagrams, descriptions, and prototype models necessary to execute the program as specified by the user. The software that is provided compliments the hardware and provides the execution of logic and arithmetic functions under program control. These large categories will be discussed in detail.